TANDBERG MPS API

User Guide

TANDBERG
D13639 Rev 06
November 2007

Table of Contents

User Guide 1
1 The TANDBERG API 1
1.1 Introduction to XML 2
1.2 Introduction to XML Path Language (XPath) 5
1.3 The TANDBERG XML Engine 7
1.4 The XML Documents 9
1.5 Introduction to TANDBERG XML API Service (TXAS) 14
1.6 Exercises 16
2 The XML-based Advanced Command Line Interface 18
2.1 XACLI 19
2.2 The Status-type root commands – xstatus / xhistory 22
2.3 The Configuration-type root commands - xconfiguration/xdirectory 24
2.4 The Command-type root commands - xcommand 27
2.5 XML Output - xgetxml 30
2.6 Special Commands 31
3 API - Configurations 37
3.1 configuration.xml – xconfiguration 38
3.2 directory.xml – xdirectory 47
4 API - Commands 48
4.1 command.xml – xcommand 49
5 API - Status 64
5.1 status.xml – xstatus 65
5.2 history.xml – xhistory 80
5.3 Event.xml – xevent 81

1 The TANDBERG API

This document is a guide to the API interface of the TANDBERG MPS products. All rights reserved. This document contains information that is proprietary to TANDBERG. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronically, mechanically, by photocopying, or otherwise, without the prior written permission of TANDBERG. Nationally and internationally recognized trademarks and trade names are the property of their respective holders and are hereby acknowledged.

Disclaimer

The information in this document is furnished for informational purposes only, is subject to change without prior notice, and should not be construed as a commitment by TANDBERG. The information in this document is believed to be accurate and reliable; however TANDBERG assumes no responsibility or liability for any errors or inaccuracies that may appear in this document, nor for any infringements of patents or other rights of third parties resulting from its use. No license is granted under any patents or patent rights of TANDBERG.

This document was written by the Research and Development Department of TANDBERG, Norway. We are committed to maintaining a high level of quality in all our documentation. Towards this effort, we welcome your comments and suggestions regarding the content and structure of this document. Please fax or mail your comments and suggestions to the attention of:

Research and Development Department
TANDBERG
Philip Pedersen vei 22
1366 Lysaker
Norway
Tel: +47 67 125 125
Fax: +47 67 125 234

COPYRIGHT © 2007, TANDBERG

1.1 Introduction to XML

XML is a markup language for documents containing structured information.

All information elements in an XML document are marked by a tag and a corresponding end-tag. The end-tag has the same name as the tag, but is prefixed with a slash, "/". All tags are put within angular brackets ("< >").

Example 1.1

Below is an example of how configurations of a Serial Port could be represented using XML.

```
<Configuration>
  <SerialPort item="1">
    <BaudRate item="1">9600</BaudRate>
    <Parity item="1">None</Parity>
    <DataBits item="1">8</DataBits>
    <StopBits item="1">1</StopBits>
    <Mode item="1">Control</Mode>
  </SerialPort>
</Configuration>
```

From the tree structure of this example we can see that `BaudRate`, `Parity`, `Databits`, `StopBits` and `Mode` are properties of the `SerialPort`. We can distinguish between *container-elements* and *value-elements*. Container-elements contain one or more sub-elements, while value-elements contain a value. This is analogous to files and folders on a computer. Container-elements are folders that can contain sub-folders and files, while value-elements are files containing data.
In the XML structure for the Serial Port we see that the container-element `SerialPort` contains five sub-elements. All these sub-elements are value-elements, each holding values for the properties: `BaudRate`, `Parity`, `DataBits`, `StopBits` and `Mode`.

Example 1.2

In this example we will look at element attributes. Attributes are used to add meta information to an element. Attributes are placed within the start tag of an element and different attributes are separated by space.

An XML structure representing the status of a call in a videoconferencing system is shown below:

```
<Status>
  <Call item="1" status="Disconnected" type="NA" protocol="NA">
    <Cause item="1">255</Cause>
  </Call>
</Status>
```

We can see from the `status` attribute of the `Call` element that the call is disconnected. The only relevant information regarding this call is the disconnect cause value. Therefore the sub-structure of the call element contains only one value-element.

Example 1.3

If we now look at the call element for an active call we see that `call` element contains a large sub-structure:

```
<Status>
  <Call item="1" status="Synced" type="Vtlph" protocol="H323">
    <CallRate item="1">768</CallRate>
    <RemoteNumber item="1">10.47.15.127</RemoteNumber>
    <Channels item="1" type="Incoming">
      <Audio item="1" status="Active">
        <Protocol item="1">G722</Protocol>
        <Rate item="1">64</Rate>
      </Audio>
      <Video item="1" status="Active">
        <Protocol item="1">H263</Protocol>
        <Resolution item="1">CIF</Resolution>
        <Rate item="1">704</Rate>
      </Video>
      <Video item="2" status="Inactive" />
      <Data item="1" status="Inactive" />
    </Channels>
    <Channels item="2" type="Outgoing">
      <Audio item="1" status="Active">
        <Protocol item="1">G722</Protocol>
        <Rate item="1">64</Rate>
      </Audio>
      <Video item="1" status="Active">
        <Protocol item="1">H264</Protocol>
        <Resolution item="1">SIF</Resolution>
        <Rate item="1">704</Rate>
      </Video>
      <Video item="2" status="Inactive" />
      <Data item="1" status="Inactive" />
    </Channels>
  </Call>
</Status>
```

In this example, the attributes are used to provide valuable information in addition to establishing a dependency to the underlying sub-structure of the element.

Example 1.4

In the above examples, all elements are having an attribute named *item*. This attribute specifies the instance number of the element. If we expand Example 1.1 to a system having two serial ports, the XML structure could look like this:

```
<Configuration>
  <SerialPort item="1">
    <BaudRate item="1">9600</BaudRate>
    <Parity item="1">None</Parity>
    <DataBits item="1">8</DataBits>
    <StopBits item="1">1</StopBits>
    <Mode item="1">Control</Mode>
  </SerialPort>
  <SerialPort item="2">
    <BaudRate item="1">19200</BaudRate>
    <Parity item="1">None</Parity>
    <DataBits item="1">8</DataBits>
```

```
    <StopBits item="1">1</StopBits>
    <Mode item="1">Auto</Mode>
  </SerialPort>
</Configuration>
```

1.2 Introduction to XML Path Language (XPath)

XPath is a comprehensive language to address data in XML documents.
It is though very simple to understand the basics. If you are able to specify the path to a file on your computer, you are able to specify the path to an element in a XML structure.

Example 1.5

Let us go back to the serial port configurations of Example 1.1.

```
<Configuration>
  <SerialPort item="1">
    <BaudRate item="1">9600</BaudRate>
    <Parity item="1">None</Parity>
    <DataBits item="1">8</DataBits>
    <StopBits item="1">1</StopBits>
    <Mode item="1">Control</Mode>
  </SerialPort>
</Configuration>
```

To specify the path to the `SerialPort` element we simply start at the root level and separate the levels in the tree structure by a slash (“/”):
`Configuration/SerialPort`

The path to the `BaudRate` element is:
`Configuration/SerialPort/BaudRate`

Example 1.6

To address a specific item of an element, the item number is added within brackets (“[]”) after the element name.
The path to the `BaudRate` element of `SerialPort` item 2 in Example 1.4 is:
`Configuration/SerialPort[2]/BaudRate`
If the item number is omitted for an element, all items of this element will be addressed. The following expression addresses the `BaudRate` element of both serial ports:
`Configuration/SerialPort/BaudRate`

Example 1.7

When using XPath it is possible to omit specifying intermediate levels in the address expression. By using the powerful “double slash” you can address elements without having to specify the complete path.
The expression below addresses the `BaudRate` element of both serial ports of Example 1.4:
`Configuration//BaudRate`

Example 1.8

XPath also supports addressing by putting constraints on element attributes. Let's go back to the `Call` element in Example 1.2. The below expression will address the `CallRate` element of all *Synced* calls in a system:

```
Status/Call[@status="Synced"]/CallRate
```

To add more constraints on element attributes, XPath supports boolean expressions. To address all *Synced H323* calls in a system, the following expression can be used:

```
Status/Call[@status="Synced" AND @protocol="H323"]/CallRate
```

1.3 The TANDBERG XML Engine

The TANDBERG XML engine is optimized for advanced machine-machine interaction between a TANDBERG system and an external control application. The main features can be summarized to:

- Structuring of information
- Addressing using XPath
- Feedback

1.3.1 Structuring of Information

An application programming interface can be seen as a gate where information is exchanged between two systems - a control application and a target system. The control application transmits instructions to the target system, while the target system supplies information about how these instructions are executed, in addition to other system related information.
Thus, the exchange of information can be divided into:

1. information flowing from target, hereby called *read information (r)*
2. information flowing to target, hereby called *write information (w)*

If we now look at the TANDBERG systems we can identify three main types of information, either being *read information (r), write information (w) or read-write information (rw):*

1. *(r) Read information* – ***Status Information***.
   Information about the system and system processes, i.e. information generated by the system.
   F.ex. status about ongoing calls, network status, conference status etc.
   All status information is structured in a hierarchy, making up a database constantly beeing updated by the system to reflect process changes.
2. *(w) Write information* – ***Command Information***.
   Information supplied by the user to initiate an action.
   F.ex. instructing the system to place a call, assigning floor to a specific site, disconnecting a site etc.
   A command is usually followed by a set of parameters to specify how the given action is to be executed.
3. *(rw) Read-Write information* – ***Configuration Information***. Information defining system settings. This information can both be supplied and read by the user. F.ex. default callrate, baudrate of a serial port, enabling/disabling of various features etc.
   All configuration information is structured in a hierarchy making up a database of system settings. But for the Configuration information, the data in the database can only be updated by the user/control application.

1.3.2 Addressing using XPath

To address information in the hierarchic structure of Status and Configuration information the TANDBERG systems support abbreviated XML Path Language (XPath). This allows the user/control application to address everything from a single element of data, f.ex. the callrate of a specific call, to larger parts of the hierarchy, f.ex. all information available for a given call.

The structuring of information together with XPath for addressing makes up powerful features like searching and setting of multiple instances of a configuration.

1.3.3 Feedback

Feedback is an extremely powerful feature where the TANDBERG system actively returns updated status and configuration information to the user/control application whenever changes occur. The user/control application can specify what parts of the status and

configuration hierarchies it wants to monitor by using XPath. The user/control application can therefore limit the amount of information it receives from the target system to only those parts being of interest for the given application.

1.4 The XML Documents

1.4.1 Documents

The XML Data in the TANDBERG systems are divided into three main types of documents. The division is based on whether the information is *Read Information*, *Write Information* or *Read-Write* information:

1. ***Status documents (r)***: Documents holding all available Status Information in the system.
   Supported documents:
   a. status.xml
   b. history.xml
2. ***Configuration documents (rw)***: Documents holding all system configurations.
   Supported documents:
   a. configuration.xml
   b. directory.xml
3. ***Command documents (w)***: Documents defining the supported system commands used to initiate system processes. This is *write* data, i.e. the parameter values for a given command are defined by the user and posted to the system. The posted values will not be returned when reading the document from the system. Reading a command document from the system returns descriptions of the supported commands with empty parameter values.
   Supported documents:
   a. command.xml
4. ***Meta Documents***: Meta documents contain information that can be referenced by other documents, e.g. value domains of configurations or command parameters.
   Supported Meta Documents:
   a. valuespace.xml

1.4.2 Status Documents (r)

The Status Documents are characterised by an extensive use of XML attributes. In addition to holding information, the attributes are used to reflect the structure of the sub-elements, which are dependent on the state of the system.

Example 9

The element `Call` will contain different sub elements depending on the call state, call type or direction:

```
<Call item="1" status="Synced" type="Vtlph" protocol=”H323”
direction="Outgoing">
  <CallRate item="1">768</CallRate>
  <RemoteNumber item="1">58458</RemoteNumber>
  <Mute item="1">Off</Mute>
  <Microphone item="1">Off</Microphone>
  <Duration item="1">15</Duration>
  <Channels item="1" type="Incoming">
    <Rate item="1">768</Rate>
    <Restrict item="1">Off</Restrict>
    <Encryption item="1" status="Off" />
    <Audio item="1" status="Active">
      <Protocol item="1">G722</Protocol>
      <Rate item="1">64</Rate>
```

```
      <RemoteIPAddress item="1" />
      <LocalIPAddress item="1">10.47.8.41:2326</LocalIPAddress>
      <Encryption item="1" status="On">
        <Type item="1">AES-128</Type>
      </Encryption>
      <RSVP item="1">Off</RSVP>
     <RSVPRate item="1">0</RSVPRate>
      <DynamicRate item="1">64</DynamicRate>
      <TotalPackets item="1">367</TotalPackets>
      <PacketLoss item="1">0</PacketLoss>
      <Jitter item="1">0</Jitter>
    </Audio>
    .
    .
    .
</Call>

---

<Call item="2" status="Synced" type="Vtlph" protocol="H320"
direction="Outgoing">
  <CallRate item="1">384</CallRate>
  <Bonding item="1">On</Bonding>
  <RemoteNumber item="1">8796</RemoteNumber>
  <RemoteNumber item="2" />
  <RemoteSubAddress item="1" />
  <Mute item="1">Off</Mute>
  <Microphone item="1">Off</Microphone>
  <LogTag item="1">25</LogTag>
  <Channels item="1" type="Incoming">
    <Rate item="1">384</Rate>
    <Restrict item="1">Off</Restrict>
    <Encryption item="1" status="Off" />
    <Audio item="1" status="Active">
      <Protocol item="1">G722</Protocol>
      <Rate item="1">56</Rate>
    </Audio>
    .
    .
    .
</Call>
---
<Call item="6" status="Disconnected" type="NA" protocol="NA"
direction="NA">
  <Cause item="1">255</Cause>
</Call>
```

In the above example we see that the `Bonding` element, `RemoteNumber[2]` and `SubAddress` is not present for H323 calls. On the other hand, for H323 calls, the `Audio` channel element holds information regarding packet loss etc., which is not present for H320 calls. If the call is disconnected, the `Call` element only contains the disconnect cause value.

1.4.3 Configuration documents (rw)

The structure of the Configuration documents is independent of system state, i.e. the structure will be constant in time. In addition to holding the values for the various configurations, each configuration value-element includes an attribute, `valueSpaceRef`, referencing the value domain for the configuration.

Example 10

From the XML structure below we see that the `BaudRate` element of `SerialPort[1]` is configured to *9600*. The `BaudRate` element references the `SerialPortBaudrate` element in the `ValueSpace` document, showing the value domain for this configuration.

```
<Configuration>>
  <SerialPort item="1">
    <BaudRate item="1"
valueSpaceRef="/ValueSpace/SerialPortBaudrate[@item='1']">9600</BaudR
ate>
     .
     .
  </SerialPort>
  .
  .
</Configuration>

---

<ValueSpace>
  <SerialPortBaudrate item="1" type="Literal">
    <Value>1200</Value>
    <Value>2400</Value>
    <Value>4800</Value>
    <Value>9600</Value>
    <Value>19200</Value>
    <Value>38400</Value>
    <Value>57600</Value>
    <Value>115200</Value>
  </SerialPortBaudrate>
</ValueSpace>
```

To change configurations, the part(s) of the document containing the configurations to be updated should be posted back to the system with the new values. This will be described thoroughly in a later section.

1.4.4 Command documents (w)

Command documents contain descriptions of the supported commands for the system. A Command consists of a Command name and a set of Command parameters. The parameter elements have attributes to denote whether the parameter is optional or required, in a addition to referencing the value domain for the given parameter.
Command parameters do not contain any values when read from the system.

Example 11

The command Dial is defined to take five parameters, while only the `Number` parameter is required as specified by the attribute `required`. The value domain for the parameters is referenced by the attribute `valueSpaceRef`.

```
<Command>
  <Dial item="1">
    <Number item="1" required="True"
valueSpaceRef="/ValueSpace/RemoteNumber"/>
    <SubAddress item="1" required="False"
valueSpaceRef="/ValueSpace/SubAddress"/>
```

```
    <CallRate item="1" required="False"
valueSpaceRef="/ValueSpace/Bandwidth"/>
    <Restrict item="1" required="False"
valueSpaceRef="/ValueSpace/OnOff"/>
    <NetProfile item="1" required="False"
valueSpaceRef="/ValueSpace/NetprofileRef"/>
  </Dial>
</Command>
```

To issue a command, the command structure is posted back to the system together with values for the various parameters. Optional parameters can be omitted when posting the structure back to the system.

Example 12

To place a call to number 999 the user can simply post the following XML structure to the system:

```
<Command>
  <Dial item="1">
    <Number item="1">999</Number>
  </Dial>
</Command>
```

When issuing Commands, the system will return an XML structure in response. The response structure will have the same name as the command issued, but it will be post fixed with "Result". All commands will have an attribute named `status`, stating whether the command was accepted or not. If a command is not accepted, the response structure will contain a cause code. If the command is accepted, the response structure may contain information relevant for the specific command.

Example 13

The Dial command in the above example may return the following response structure:

```
<Command>
  <DialResult item="1" status="OK">
    <CallRef item="1">1</CallRef>
    <LogTag item="1">6</LogTag>
  </DialResult>
</Command>
```

The response structure for the Dial command, `DialResult`, states that the command was accepted by the system. In addition to stating that the command was accepted, the Dial command returns the elements `CallRef` and `LogTag`. This lets the user identify/trace the call in the Status documents (status.xml and history.xml).

Example 14

Below is an example of the Dial command, not being accepted by the system:

```
<Command>
  <DialResult item="1" status="Error">
    <Cause item="1">17</Cause >
    <Description item="1">Too much bandwidth requested</Description >
  </DialResult>
</Command>
```

1.5 Introduction to TANDBERG XML API Service (TXAS)

TXAS is a service provided by TANDBERG units for transmitting and receiving (transceiving) information encoded in XML format.
The API uses HTTP(S) as the transport mechanism and connects to the normal web port (80). TXAS can be accessed in two ways; bare-bone HTTP requests where URL's uniquely identifies the request, and SOAP where a single URI is used but the request itself is encoded with XML.

1.5.1 Bare-bone HTTP(S) access

The bare-bone HTTP mode uses a unique URL to identify the specific request. The contents of the HTTP body will be a XML document (or part of it).
Bare-bone HTTP(S) access is accomplished by passing arguments in the query string (after '?' in URL) in a GET request, or using the "application/x-www-form-urlencoded" content-type mehtod of POSTing form data (Each argument starts with a name '=' and a value, and every parameter separated with '&' (and opt NL).)

getxml

REQUEST:
  /getxml
PARAM:
  location = XPath expression

"/getxml" request returns an XML document based on the location parameter passed to the request. The elements (or complete document) matching the expression will be returned.
On Incorrect XPath expression, a <Fault> element with a <XPathError> element will be returned.

formputxml

REQUEST:
  /formputxml
PARAM:
  xmldoc   = "an XML document of Configuration, Directory or Command"

This is most useful in a POST (to extend character limit of 255 of GET urls). It posts a Configuration or Command document to set the configurations or issue a command.
Like getxml, it has the data URL form-data encoded with one single parameter. The Content-Type of the document must be of type "application/x-www-form-urlencoded" and the body must be encoded accordingly (e.g. first line will be xmldoc=<then the document>).

putxml

REQUEST:
  /putxml
PARAM:
  HTTP BODY as argument

Putxml is like "formputxml", put uses the complete BODY as argument (i.e. the content of the xmldoc parameter). The Content-type should be "text/xml" or "application/xml" (or "text/plain"), though no check at the moment. (Except for application/x-www-form-urlencoded, this will cause a failure).

1.5.2 SOAP

SOAP is described in a separate section.

1.6 Exercises

The exercises in this section are based on using a TANDBERG 6000 MXP codec and Microsoft Internet Explorer. Some of the examples may however also apply to other systems and other browsers.

NOTE! Replace the ip address, 10.47.8.41, in the below examples with the ip address of your system.

Exercise 1
The example in this exercise shows how to read the supported XML documents from the system using a web browser.
Enter the following address in the browsers address field:

http://10.47.8.41/status.xml
http://10.47.8.41/history.xml
http://10.47.8.41/configuration.xml
http://10.47.8.41/directory.xml
http://10.47.8.41/command.xml
http://10.47.8.41/valuespace.xml

Exercise 2
This exercise shows how to use *getxml* to read the supported XML documents from the system. Enter the following expressions in the browsers address field (NOTE! The first letter in the document names is uppercase):

http://10.47.8.41/ getxml?location=Status
http://10.47.8.41/ getxml?location=History
http://10.47.8.41/ getxml?location=Configuration
http://10.47.8.41/ getxml?location=Directory
http://10.47.8.41/ getxml?location=Command
http://10.47.8.41/ getxml?location=ValueSpace

Exercise 3
This exercise shows how to use XPath expressions to read subsets of the XML documents.

http://10.47.8.41/getxml?location=Status/SystemUnit
http://10.47.8.41/getxml?location=Configuration/SerialPort/BaudRate
http://10.47.8.41/getxml?location=ValueSpace/SerialPortBaudrate[@item='1']
http://10.47.8.41/getxml?location=Configuration//Mode
http://10.47.8.41/getxml?location=Command/Dial

Exercise 4
The address: http://10.47.8.41/xmlput.ssi contains an editor where XML data can be edited and then posted to the system by pressing the save button. Below are examples of XML structures to be posted to the system:

```
<Configuration>
  <SerialPort>
    <BaudRate>19200</BaudRate>
  </SerialPort>
</Configuration>

---
```

```
<Configuration>
  <SerialPort>
    <BaudRate>2400</BaudRate>
  </SerialPort>
  <Conference>
    <H263>Off</H263>
    <Downspeed>Off</Downspeed>
  </Conference>
</Configuration>

---

<Command>
  <Dial>
    <Number>10.47.8.42</Number>
  </Dial>
</Command>

---

<Command>
  <DisconnectCall/>
</Command>
```

2 The XML-based Advanced Command Line Interface

The XML-based Advanced Command Line Interface, XACLI, is a very flexible interface both optimized for machine-machine interaction and man-machine interaction. It is based on the powerful TANDBERG XML engine and offers many of the same features as the TANDBERG XML interface.

The main distinction between XACLI and the TANDBERG XML interface is the input format. As XACLI is a command line interface all inputs from the user/control application have to be put on one line, in opposite to the XML interface where a complete XML document can be posted to the system in one operation.

A basic understanding of the information structuring in the TANDBERG XML engine is important in order to get the most out of the XACLI interface. It is therefore recommended to read the documentation of the TANDBERG XML API prior to reading this section.

2.1 XACLI

2.1.1 Accessing XACLI

XACLI can be accessed through Telnet via the LAN interface or through RS-232 by connecting a serial cable to the serial interface connector, referred to as the *Dataport*.
48 Telnet sessions can be active at the same time in addition to the RS-232 connection.

2.1.2 Root commands

For each of the XML documents supported by the system, there is a corresponding XACLI root command. The root command has the same name as the corresponding XML document, except that the root command is prefixed by an “x”:

XML document	XACLI root command
status.xml	xstatus
history.xml	xhistory
configuration.xml	xconfiguration
directory.xml	xdirectory
command.xml	xcommand

The information in the TANDBERG XML engine is divided into three main types: *Status Information, Configuration Information* and *Command Information*, ref. the documentation of the TANDBERG XML API.

As there is a fundamental difference in these three main types of information, there is also three different ways of working with the information using XACLI.

2.1.3 Addressing

XACLI supports XPath for addressing Status Information and Configuration Information.
In addition there is support for the proprietary TANDBERG SimplePath notation. With SimplePath notation an element or a group of elements are addressed by supplying a space separated list of element names (elemName) and optional element instance numbers (item):

<elemName> [item] <elemName> [item] ...

If the instance number of a given element is omitted, the expression addresses all instances of this element

Example 2.1

To address the `BaudRate` sub-element of `SerialPort 2`:

XPath: SerialPort[2]/BaudRate
SimplePath: SerialPort 2 BaudRate

To address the `BaudRate` sub-element of all `SerialPort` elements:

XPath: SerialPort/BaudRate
SimplePath: SerialPort BaudRate

2.1.4 Exposure options

By adding an exposure option after the address (XPath or SimplePath) expression, the system can be instructed to return only parts of the information within an element structure.

<root command> <address expression> <exposure option>

Supported exposure options:

- "-" hides all value elements
- "--" hides all sub-elements

Example 2.2
Request for *Call 1* element with no exposure option

```
xstatus call 1

*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
     CallRate: 768
     RemoteNumber: "10.47.15.127"
     Mute: Off
     Microphone: Off
     Duration: 10
     Channels 1 (type=Incoming):
       Rate: 768
       Restrict: Off
       Encryption (status=Off): /
       Audio (status=Active):
         Protocol: G722
         Rate: 64
       Video 1 (status=Active):
         Protocol: H263
         Resolution: CIF
         Rate: 704
       Video 2 (status=Inactive): /
       Data (status=Inactive): /
     Channels 2 (type=Outgoing):
       Rate: 768
       Restrict: Off
       Encryption (status=Off): /
       Audio (status=Active):
         Protocol: G722
         Rate: 64
       Video 1 (status=Active):
         Protocol: H263+
         Resolution: ICIF
         Rate: 704
       Video 2 (status=Inactive): /
       Data (status=Inactive): /
*s/end
```

Request for *Call 1* element with exposure option "-":

```
xstatus call 1 -

*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
     Channels 1 (type=Incoming):
       Encryption (status=Off): /
```

```
       Audio (status=Active):
       Video 1 (status=Active):
       Video 2 (status=Inactive): /
       Data (status=Inactive): /
     Channels 2 (type=Outgoing):
       Encryption (status=Off): /
       Audio (status=Active):
       Video 1 (status=Active):
       Video 2 (status=Inactive): /
       Data (status=Inactive): /
*s/end
```

Request for *Call 1* element with exposure option "--":

```
xstatus call 1 --

*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
*s/end
```

2.1.5 Misc

The XACLI interface is not case sensitive.
XACLI allows using only partial names.

2.2 The Status-type root commands – xstatus / xhistory

The information accessible through these commands is the exact same information that is available in the corresponding XML documents.

To get an overview of accessible top-level elements within a status-type root command, type `?` or `help` after the status-type root command.

Example 2.3

```
xstatus ?

- Status -
 Audio                  Feedback [1..3]
 BRI [1..6]             G703
 Call [1..11]           H323Gatekeeper
 Camera [1..5]          IP
 CameraTracking         PRI
 Conference             Screensaver
 Ethernet               SystemUnit
 ExternalNetwork        VirtualMonitor [1..4]
 FarEndInformation

OK
```

To access status-type data, simply type the status-type root command (xstatus or xhistory) and then an XPath address expression or a TANDBERG SimplePath expression:

```
<status-type root command> <address expression>
```

Example 2.4

```
xstatus call 1 remotenumber

*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
     RemoteNumber: "10.47.15.127"
*s/end

OK
```

2.2.1 Format

Status information is presented by a mark-up notation, similar to XML.
Main differences:

- all braces are removed in the XACLI format
- XACLI is not using end-tags, except for a tag to mark end of top element
- XACLI is using indent spaces to present the data structure

- XACLI hides instance number (*item* number in XML) of an element if there only exist one instance of a given element
- A status top level element starts with "*s"

Example 2.5 shows XML formatting and XACLI formatting for the same status element, *IP*.

Example 2.5

XML:

```
<Status>
  <IP item="1">
    <Address item="1">10.47.8.20</Address>
    <SubnetMask item="1">255.255.248.0</SubnetMask>
    <Gateway item="1">10.47.8.1</Gateway>
  </IP>
</Status>
```

XACLI:

```
*s IP:
     Address: "10.47.8.20"
     SubnetMask: "255.255.248.0"
     Gateway: "10.47.8.1"
*s/end
```

NOTE! To write a parser for the XACLI format, the parser must keep track of the levels by counting white spaces. The indent is increased by two whitespaces for each level.

2.3 The Configuration-type root commands - xconfiguration/xdirectory

The information accessible through these commands is the exact same information that is available in the corresponding XML documents.

To get an overview of accessible top-level configuration elements, type ? or help after the configuration-type root command:

```
<configuration-type root command> ?
```

Example 2.6

```
xconfiguration ?

 - User Configurations -
 AlertSpeaker           FTP               NetProfile [1..6]
 AlertTone              G703              OptionKey
 Audio                  H320              OSD
 AutoAnswer             H323              PictureProgram [1..4]
 AutoPIP                H323CallSetup     Preset [1..15]
 Bonding                H323Gatekeeper    QoS
 CallManager            H323Prefix        RTP
 Camera [1..5]          HTTP              Screensaver
 CameraSleep            HTTPS             SelfViewOnStartup
 CameraTracking         IP                SerialPort [1..2]
 Conference             IPMedia           SNMP
 CorporateDirectory     IRControl         StillImageSource
 DoNotDisturb           ISDN              Streaming
 DualMonitor            Keyboard          SystemUnit
 DuoVideoSource         LocalLayout       T1
 E1                     Logo              Telnet
 Ethernet               LoS               TelnetChallenge
 ExternalNetwork        MainVideoSource   VideoVNC
 FECC                   NAT               VNC
OK

xdirectory ?

 - Directory -
 GlobalEntry [1..400]   LocalEntry [1..200]
 GroupEntry [1..50]
OK
```

2.3.1 Configuration help

To get help on configurations, type the configuration-type root command – then an address expression followed by ? or help. The possible values for the elements matching the address expression will be returned.

```
<configuration-type root command> <address expr> ?/help
```

Example 2.7

User wants to configure IP:

```
xconfiguration ip ?
*h xConfiguration IP Assignment: <DHCP/Static>
*h xConfiguration IP Address: <IPAddr>
*h xConfiguration IP SubnetMask: <IPAddr>
*h xConfiguration IP Gateway: <IPAddr>
```

NOTE! Only typing `xconfiguration ?,` actually addresses all configuration elements within the xconfiguration root command. One would therefore expect that help on all configurations would be returned. But as described above, this is a special case and only listings of the top level elements are returned. To get help on all configurations supported by the system, type:

`xconfiguration // ?`

or

`xconfiguration ??`

2.3.2 Configuration read

To read configurations, type the configuration-type root command followed by an address expression:

```
<configuration-type root command> <address expr>
```

Example 2.8

User wants to read IP configurations:

```
xconfiguration ip
*c xConfiguration IP Assignment: Static
*c xConfiguration IP Address: "10.47.8.20"
*c xConfiguration IP SubnetMask: "255.255.248.0"
*c xConfiguration IP Gateway: "10.47.8.1"

OK
```

2.3.3 Configuration set (write)

To set configurations, the address expression following the configuration-type root command must end with a colon. The value to be set must be added after the colon:

```
<configuration-type root command> <address expr>: value
```

Example 2.9

User wants to set IP assignment:

```
xconfiguration ip assignment: static
```

or

```
xconfiguration ip/assignment: static
```

2.4 The Command-type root commands - xcommand

To get an overview of the supported commands within a command-type root command, type ? or help after the command-type root command.

```
<command-type root command> ?
```

Example 2.10

xcommand ?

```
 - User Commands -
 Boot                  DuoVideoStart         MessageBoxDisplay
 CallAccept            DuoVideoStop          PIPHide
 CallMute              FECCFocus             PIPShow
 CameraBrightness      FECCMove              PresetActivate
 CameraFocus           FECCPresetActivate    PresetStore
 CameraHalt            FECCPresetStore       ScreensaverActivate
 CameraMove            FECCRequestStill      ScreensaverDeactivate
 CameraPosition        FECCSelectSource      ScreensaverReset
 CameraTrackingStart   FeedbackDeregister    SiteDisconnect
 CameraTrackingStop    FeedbackRegister      SiteView
 CameraWhiteBalance    FloorRelease          SiteViewEnd
 ChairRelease          FloorRequest          SPIDAutoConfigure
 ChairTake             FloorToSite           StillImageSend
 ConferenceDisconnect  FloorToSiteEnd        StreamingStart
 DefaultValuesSet      GroupEntryAdd         StreamingStop
 Dial                  GroupEntryDelete      TextDelete
 DialGlobalEntry       LocalEntryAdd         TextDisplay
 DialGroupEntry        LocalEntryDelete      VirtualMonitorReset
 DialLocalEntry        MessageBoxDelete      VirtualMonitorSet
 DisconnectCall
OK
```

To list usage for all commands with parameters, type a double question mark after the command-type root command.

```
<command root command> ??
```

Example 2.11
xcommand ??

2.4.1 Command help

To get help on a specific command, type the command-type root command – then a command name followed by ? or `help`:

```
<command-type root command> <command name> ?
```

Example 2.12

```
xcommand Dial ?

*h xCommand Dial
             Number(r): <S: 0, 30>
             SubAddress: <S: 0, 10>
             CallRate: <1xh221/2xh221/64/128/256...
             Restrict: <On/Off>
             NetProfile: <1..6>
OK
```

NOTE! Required parameters are identified by an “(r)” behind the parameter name.

2.4.2 Issuing a command

A command must start with a command-type root command, followed by a command name, followed by a set of parameters. Parameters values can either be specified by a mark-up notation or by placing the parameter values in the sequence specified by the help text – or a combination of these methods.

Markup notation

<command-type root command> <command> <parameter:value> <parameter:value>...

When using this notation, the sequence the parameters are entered is unessential:

Example 2.13

```
xcommand dial number:666 restrict:on callrate:128 subaddress:10
```

Abbreviations can be used for the parameter names as long as the parameter names are unique within the command:

Example 2.14

```
xcommand dial nu:666 r:on c:128 s:10
```

If there are multiple instances of a parameter, the item number is added after the tag separated with a dot:

<command-type root command> <command> <parameter.item:value> <parameter.item:value>...

Example 2.15

```
xcommand groupentryadd name:TANDBERG localentryid.1:15
localentryid.2:57
```

Sequence notation

<command-type root command> <command> <value> <value>...

When using this notation the parameter values must be entered in the sequence as stated in the help text:

Example 2.16

```
xcommand dial 666 10 128 on
```

Combination

A combination of mark-up notation and sequence are also supported. The marked parameters will be assigned the user entered values first, and then the system will assign the sequence entered parameters for the parameters not yet having been assigned a value:

Example 2.17

```
xcommand dial 666 r:on 10 128
```

Command response

When issuing a command, the system will return a set of return values, ref. the documentation of the TANDBERG XML API. The response will be on the same format as the standard XACLI Status format.

Example 2.18

```
xcommand dial 10.47.15.127

*r Result (status=OK):
     CallRef: 1
     LogTag: 6
*r/end
OK
```

NOTE! When using XACLI as a machine-machine interface it is recommended to use markup notation and always supply complete tag names.

2.5 XML Output - xgetxml

As an alternative to the standard XACLI output format, XML format is supported through the root command ***xgetxml***. *xgetxml* takes an XPath expression as parameter and the elements (or complete document) matching the expression will be returned.

Example 2.19

```
xgetxml status/ip

<Status>
  <IP item="1">
    <Address item="1">10.47.8.20</Address>
    <SubnetMask item="1">255.255.248.0</SubnetMask>
    <Gateway item="1">10.47.8.1</Gateway>
  </IP>
</Status>
OK
```

2.6 Special Commands

In addition to the root commands described above, XACLI support a set of root commands that only applies to the Telnet session or RS232 session from where they are issued. This lets the user/control application individually configure the session(s) in use.

Supported special commands:

- xfeedback (not supported on all platforms)
- xpreferences

2.6.1 xfeedback

The special command *xfeedback* lets the user register user defined XPath expressions (with possible *exposure options*) to monitor changes in the XML/XACLI data. Whenever there is a change in one or more elements addressed by a registered XPath expression, the part of the element structure containing these changes will be returned. The system supports a total of 20 registered expressions, with a total of 15 expressions for one session.

```
xfeedback ?

usage: xfeedback register <XPathExpression>
or:    xfeedback deregister <index>
or:    xfeedback list
-
(note: deregistration with index=0 will deregister all registered
expressions)

OK
```

Example 2.20

User wants to monitor changes in audio protocols for all active calls:

```
xfeedback register status/call/channels/audio/protocol
```

To view registered expressions:

```
xfeedback list

*xf 1 status/call/channels/audio/protocol
OK
```

The call changes audio protocol from G722 to G728 on incoming audio channel on call 1:

```
*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
     Channels 1 (type=Incoming):
       Audio (status=Active):
         Protocol: G728
*s/end
```

When changing back to G722:

```
*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
     Channels 1 (type=Incoming):
       Audio (status=Active):
         Protocol: G722
*s/end
```

Example 2.21

Exposure options are also supported together with feedback.
User only wants to monitor call setup progression.

```
xfeedback register status/call--

OK

xcom dial 10.47.15.127

*s Call 1 (status=EstablOut, type=Vtlph, protocol=H323,
direction=Outgoing):
*s/end

*r Result (status=OK):
     CallRef: 1
     LogTag: 3
*r/end

OK

*s Call 1 (status=Alerting, type=Vtlph, protocol=H323,
direction=Outgoing):
*s/end
CONNECT

*s Call 1 (status=Syncing, type=Vtlph, protocol=H323,
direction=Outgoing):
*s/end

*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
*s/end
```

Example 2.22

User only wants to know when calls are connected and disconnected:

```
xfeedback register status/call[@status="Synced"]--

OK

xfeedback register status/call[@status="Disconnected"]--

OK

xcom dial 10.47.15.127

*r Result (status=OK):
     CallRef: 1
     LogTag: 4
*r/end

OK

CONNECT

*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
```

```
*s/end

xcom disc

*r Result (status=OK): /
*r/end

OK

NO CARRIER
*s Call 1 (status=Disconnected, type=NA, protocol=NA, direction=NA):
*s/end
```

When conditional XPath expressions are used, the system will provide feedback on all elements within the address the first time the condition is true.

Example 2.23

User wants to monitor call changes only when the call is in *Synced* state. By registering the below expression, the system will not provide feedback on the call before it reaches the *Synced* state. When it first enters the *Synced* state it will provide status for the complete call. After this, the system will only give feedback on elements changing values (provided that the call is still in *Synced* state).

```
xfeedback register status/call[@status="Synced"]

OK

xcom dial 10.47.15.127

*r Result (status=OK):
     CallRef: 1
     LogTag: 5
*r/end

OK

CONNECT
*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
     CallRate: 768
     RemoteNumber: "10.47.15.127"
     Mute: Off
     Microphone: Off
     Duration: 0
     Channels 1 (type=Incoming):
       Rate: 768
       Restrict: Off
       Encryption (status=Off): /
       Audio (status=Active):
         Protocol: G722
         Rate: 64
       Video 1 (status=Active):
         Protocol: H263
         Resolution: CIF
         Rate: 704
       Video 2 (status=Inactive): /
       Data (status=Inactive): /
     Channels 2 (type=Outgoing):
```

```
       Rate: 768
       Restrict: Off
       Encryption (status=Off): /
       Audio (status=Active):
         Protocol: G722
         Rate: 64
       Video 1 (status=Active):
         Protocol: H263+
         Resolution: ICIF
         Rate: 704
       Video 2 (status=Inactive): /
       Data (status=Inactive): /
*s/end
```

…suddenly there is a change in audio protocol:

```
*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
     Channels 1 (type=Incoming):
       Rate: 704
       Audio (status=Inactive): /
*s/end
*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
     Channels 1 (type=Incoming):
       Rate: 720
       Audio (status=Active):
         Protocol: G728
         Rate: 16
*s/end
```

2.6.2 xpreferences

The special command *xpreferences* lets the user/control application individually configure the Telnet/RS-232 session in use.

`xpreferences ?`

```
usage: xpreferences xpathwrite <on/off>
or:    xpreferences detaillevel <1..2>
or:    xpreferences xmlconfigfeedback <on/off>
or:    xpreferences xmlstatusfeedback <on/off>
or:    xpreferences xmlcommandresult <on/off>

OK
```

`xpreferences xpathwrite <on/off>`
This command disables/enables the XPath engine when issuing configurations. When the XPath engine is disabled, the user/control application must supply the complete path to the configurations to be set (no “double slashes” allowed). This will improve the performance of the system when issuing many consecutive configurations.

NOTE! It is always recommended to supply the complete path for configurations to be set when issuing commands from an external control application.

`xpreferences detaillevel <1..2>`
Most information elements accessible by the status-type root commands are defined to be level1 information. However there are some information elements which are defined to be level2 information. When reading status information, only the information elements with a detail level equal to or less than the `detaillevel` defined for the interface will be listed.

Example 2.24

```
xstat call 1 channels 1 audio

*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
     Channels 1 (type=Incoming):
       Audio (status=Active):
         Protocol: G722
         Rate: 64
*s/end

OK

xpreferences detaillevel 2

OK

xstat call 1 channels 1 audio

*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing, logTag=3):
     Channels 1 (type=Incoming):
       Audio (status=Active):
         Protocol: G722
         Rate: 64
         RemoteIPAddress: ""
         LocalIPAddress: "10.47.8.28:2326"
         Encryption (status=On):
           Type: AES-128
         RSVP: Off
       RSVPRate: 0
         DynamicRate: 64
         TotalPackets: 1618
       PacketLoss: 0
         Jitter: 0
*s/end

OK
```

`xpreferences xmlconfigfeedback <on/off>`
If *xmlconfigfeedback* is set to on, feedback on configurations will be returned in XML-format instead of the standard XACLI configuration format.

Example 2.25

```
XACLI-format:
*c xConfiguration SerialPort 1 BaudRate: 2400

XML-format:
<Configuration>
  <SerialPort item="1">
```

```
    <BaudRate item="1">2400</BaudRate>
  </SerialPort>
</Configuration>
```

`xpreferences xmlstatusfeedback <on/off>`
If *xmlstatusfeedback* is set to on, all status feedback will be returned in XML-format instead of the standard XACLI status format.

Example 2.26

```
XACLI-format:
*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing):
     Channels 1 (type=Incoming):
       Rate: 736
       Audio (status=Active):
         Protocol: G722_1
         Rate: 32
*s/end

XML-format:
<Status>
  <Call item="1" status="Synced" type="Vtlph" protocol="H323"
direction="Outgoing">
    <Channels item="1" type="Incoming">
      <Rate item="1">768</Rate>
      <Audio item="1" status="Active">
        <Protocol item="1">G722</Protocol>
        <Rate item="1">64</Rate>
      </Audio>
    </Channels>
  </Call>
</Status>
```

`xpreferences xmlcommandresult <on/off>`
If *xmlcommandresult* is set to on, response for commands will be returned in XML-format.

Example 2.27

```
XACLI-format:
xcom dial 10.47.15.127

*r Result (status=OK):
     CallRef: 1
     LogTag: 8
*r/end

XML-format:
xcom dial 10.47.15.127

<Result item="1" status="OK">
  <CallRef item="1">1</CallRef>
  <LogTag item="1">8</LogTag>
</Result>
```

3 API - Configurations

This section gives an overview of the Configuration Information available in the Configuration XML documents (*configuration.xml / directory.xml*) and the Configuration root commands (*xconfiguration / xdirectory*) of the XACLI interface.

All examples are presented using the standard XACLI format.

3.1 configuration.xml – xconfiguration

AllowIncomingTlphCall	**AllowIncomingTlphCall: <On/Off>**
Conference	**Conference [1..35] Numbers E164Alias: <E164: 0, 31>** **Conference [1..35] Numbers PRI: <S: 0, 60>** **Conference [1..35] Numbers H323ID: <S: 0, 50>** **Conference [1..35] Numbers SIP URI: <S: 0, 60>**
ConferenceTemplate	**ConferenceTemplate [1..10] Name: <S: 0, 30>** **ConferenceTemplate [1..10] CallRate: <1xh221/2xh221/64/128/192/256/320/384/H0/512/768/1152/1472/1536/1920/Tlph>** **ConferenceTemplate [1..10] Restrict: <On/Off>** **ConferenceTemplate [1..10] MaxVideoSites: <0..160>** **ConferenceTemplate [1..10] MaxAudioSites: <0..16>** **ConferenceTemplate [1..10] Password: <S: 0, 8>** **ConferenceTemplate [1..10] PasswordOnOutgoingCalls: <On/Off>** **ConferenceTemplate [1..10] Encryption: <On/Off>** **ConferenceTemplate [1..10] EncryptionType: <DES/AES-128/Auto>** **ConferenceTemplate [1..10] PictureMode: <Auto/VS/2Split/3Split/4Split/4+3Split/5+1Split/7+1Split/8+2SplitTopBottom/8+2SplitBottomTop/8+2SplitTop/8+2SplitBottom/9Split/12+1SplitCenter/12+1SplitTopLeft/12+2Split/16Split/30+2Split/2+1SplitWide/3SplitWide/3+1SplitWide/4+1SplitWide/6SplitWide/8+1SplitWide/12SplitWide/CPAuto>** **ConferenceTemplate [1..10] VideoFormat: <Auto/Motion/Sharpness>** **ConferenceTemplate [1..10] CustomFormats: <On/Off>** **ConferenceTemplate [1..10] AGC: <On/Off>** **ConferenceTemplate [1..10] TelephoneFilter: <On/Off>** **ConferenceTemplate [1..10] AllowIncomingCalls: <On/Off>** **ConferenceTemplate [1..10] Duration: <0..999>**

ConferenceTemplate [1..10] EntryExitTones: <On/Off>

ConferenceTemplate [1..10] WelcomeMessage: <On/Off>

ConferenceTemplate [1..10] LegacyLevel: <0..15>

ConferenceTemplate [1..10] DuoVideo: <On/Off>

ConferenceTemplate [1..10] AudioG728: <On/Off>

ConferenceTemplate [1..10] CascadingPreference: <Auto/Master/Slave>

ConferenceTemplate [1..10] BillingCode: <S: 0, 16>

ConferenceTemplate [1..10] CPAutoSwitch: <0..60>

ConferenceTemplate [1..10] FloorToFull: <On/Off>

ConferenceTemplate [1..10] WebCallListTimeout: <On/Off>

ConferenceTemplate [1..10] NetworkId: <1..32>

ConferenceTemplate [1..10] ConferenceSelfview: <On/Off>

ConferenceTemplate [1..10] PhoneIndication: <On/Off>

ConferenceTemplate [1..10] SpeakerIndication: <On/Off>

ConferenceTemplate [1..10] VideoText: <On/Off/Auto>

ConferenceTemplate [1..10] VideoTextTimeout: <0..30>

ConferenceTemplate [1..10] ChairControl: <On/Off>

ConferenceTemplate [1..10] LectureMode: <On/Off>

ConferenceTemplate [1..10] Protect: <On/Off>

ConferenceTemplate [1..10] BandwidthThreshold: <64/128/192/256/320/384/512/768/1152/1472/1920>

ConferenceTemplate [1..10] NetErrorHandling: <IPLR/FURBlock/None>

ConferenceTemplate [1..10] IPLRRobustMode: <Auto/On>

ConferenceTemplate [1..10] FURBlockSites: <Auto/On>

ConferenceTemplate [1..10] FURFilterInterval: <0..60>

ConferenceTemplate [1..10] HDEnabled: <On/Off>

ConferenceTemplate [1..10] VoiceSwitchTimeout: <1..10>

ConferenceTemplate [1..10] SecondaryRate: <On/Off>

ConferenceTemplate [1..10] FarTlphEchoSupression:

	<Off/Normal/High> **ConferenceTemplate [1..10] OptimalVideoQuality: <On/Off>** **ConferenceTemplate [1..10] EncoderSelectionPolicy: <BestBitRate/BestVideoStandard/BestResolution>** **ConferenceTemplate [1..10] BandwidthManagement: <Manual/Auto>** **ConferenceTemplate [1..10] WebSnapshots: <On/Off>** **ConferenceTemplate [1..10] AutoAspectRatio4x3: <On/Off>**
Ethernet	**Ethernet [1..2] Speed: <Auto/10half/10full/100half/100full/None>**
ExternalManager	**ExternalManager Address: <S: 0, 64>** **ExternalManager Path: <S: 0, 255>**
FeedbackFilter	**FeedbackFilter Conference: <0..10>** **FeedbackFilter Call: <0..10>**
Gateway	**Gateway BandwidthPercentage: <0..100>** *** Gateway LoadLimit: <0..100>** **Gateway NaturalVideo: <On/Off>** **Gateway CustomFormats: <On/Off>** **Gateway DuoVideo: <On/Off>** **Gateway Encryption Mode: <Independent/Transparent>** **Gateway Encryption H320: <On/Off/Auto>** **Gateway Encryption H323: <On/Off/Auto>** **Gateway H264: <On/Off>** **Gateway H264rcdo: <On/Off>** **FarTlphEchoSupression: <Off/Normal/High>**
H323CallSetup	**H323CallSetup [1..2] Mode: <Direct/Gatekeeper>**
H323Gatekeeper	**H323Gatekeeper [1..2] Address: <S: 0, 64>** **H323Gatekeeper [1..2] Authentication Mode: <Auto/Off>** **H323Gatekeeper [1..2] Authentication ID: <S: 0, 50>**

	H323Gatekeeper [1..2] Authentication Password: <S: 0, 50>
HTTP	**HTTP Mode: <On/Off>**
HTTPS	**HTTPS Mode: <On/Off>**
IPProtocol	**IPProtocol: <IPv4/IPv6/Both>**
IP	**IP [1..2] Address: <S: 0, 15>** **IP [1..2] SubnetMask: <S: 0, 15>** **IP [1..2] Gateway: <S: 0, 15>** **IP [1..2] DNS Server [1..5] Address: <IPv4v6Addr: 0, 43>** **IP [1..2] DNS Domain Name: <S: 0, 64>** **IP [1..2] Description: <S: 0, 16>**
ISDNInterfaceCard	**ISDNInterfaceCard [1..6] NetType: <PRI/G703>** **ISDNInterfaceCard [1..6] ISDN SendComplete: <On/Off>** **ISDNInterfaceCard [1..6] ISDN SendNumber: <On/Off>** **ISDNInterfaceCard [1..6] ISDN ParallelDial: <On/Off>** **ISDNInterfaceCard [1..6] ISDN HLC: <On/Off>** **ISDNInterfaceCard [1..6] ISDN SpeechTimers: <On/Off>** **ISDNInterfaceCard [1..6] ISDN PRI NSFTelephony Mode: <On/Off>** **ISDNInterfaceCard [1..6] ISDN PRI NSFTelephony Number: <0..31>** **ISDNInterfaceCard [1..6] ISDN PRI NSFVideoTelephony Mode: <On/Off>** **ISDNInterfaceCard [1..6] ISDN PRI NSFVideoTelephony Number: <0..31>** **ISDNInterfaceCard [1..6] ISDN PRI SwitchType: <NI/ATT/Euro/Japan>** **ISDNInterfaceCard [1..6] ISDN PRI TrunkGroups: <On/Off>** **ISDNInterfaceCard [1..6] ISDN PRI InitialRestart: <On/Off>** **ISDNInterfaceCard [1..6] ISDN PRI L2WindowSize: <1..7>** **ISDNInterfaceCard [1..6] ISDN PRI Alert: <On/Off>** **ISDNInterfaceCard [1..6] ISDN PRI ChanId: <On/Off>**

	ISDNInterfaceCard [1..6] ISDN PRI Interface [1..8] MaxChannels: <1..30> **ISDNInterfaceCard [1..6] ISDN PRI Interface [1..8] HighChannel: <1..31>** **ISDNInterfaceCard [1..6] ISDN PRI Interface [1..8] LowChannel: <1..31>** **ISDNInterfaceCard [1..6] ISDN PRI Interface [1..8] Search: <High/Low>** **ISDNInterfaceCard [1..6] ISDN PRI Interface [1..8] NumberRangeStart: <S: 0, 60>** **ISDNInterfaceCard [1..6] ISDN PRI Interface [1..8] NumberRangeStop: <S: 0, 60>** **ISDNInterfaceCard [1..6] ISDN PRI Interface [1..8] Enable: <On/Off>** **ISDNInterfaceCard [1..6] ISDN PRI Interface [1..8] NFASEnable: <On/Off>** **ISDNInterfaceCard [1..6] ISDN PRI Interface [1..8] NFASGroupId: <1..4>** **ISDNInterfaceCard [1..6] ISDN PRI Interface [1..8] NFASInterfId: <0..127>** **ISDNInterfaceCard [1..6] G703 PhysicalLayer: <E1/T1>** **ISDNInterfaceCard [1..6] G703 Interface [1..8] Enable: <On/Off>** **ISDNInterfaceCard [1..6] G703 Interface [1..8] IncludeE1Channel16: <On/Off>** **ISDNInterfaceCard [1..6] G703 Interface [1..8] Call [1..5] StartChannel: <0..30>** **ISDNInterfaceCard [1..6] G703 Interface [1..8] Call [1..5] MaxChannels: <0..30>** **ISDNInterfaceCard [1..6] E1 Port [1..8] CRC4: <On/Off>** **ISDNInterfaceCard [1..6] T1 Port [1..8] CableLength: <Range1/Range2/Range3/Range4/Range5>** **ISDNInterfaceCard [1..6] Description: <S: 0, 16>**
LoS	**LoS Duration Exponent: <10..30>** **LoS Duration Offset: <0..65535>** **LoS Inhibit: <0..65535>** **LoS Initial: <0..65535>**

	LoS Polarity: <Positive/Negative> **LoS Retry: <0..65535>**
MCU	**MCU H264: <On/Off>** **MCU h264rcdo: <On/Off>**
MediaBoard	**MediaBoard [1..12] IP Address: <IPAddr>** **MediaBoard [1..12] IP SubnetMask: <IPAddr>** **MediaBoard [1..12] IP Gateway: <IPAddr>** **MediaBoard [1..12] IP NetworkId: <1..2>** **MediaBoard [1..12] Ethernet Speed: <Auto/10half/10full/100half/100full/None>** **MediaBoard [1..12] Description: <S: 0, 16>**
NetProfile	**NetProfile [1..7] Name: <S: 0, 8>** **NetProfile [1..7] CallPrefix: <S: 0, 9>** **NetProfile 1 Network: <Auto>** **NetProfile 2 Network: <H320>** **NetProfile 3 Network: <H323>** **NetProfile 4 Network: <H320/H323/SIP/Auto>** **NetProfile 5 Network: <H320/H323/SIP/Auto>** **NetProfile 6 Network: <H320/H323/SIP/Auto>** **NetProfile 7 Network: <SIP>**
NTP	**NTP Address: <IPv4v6Addr: 0, 43>**
PrefixDialIn	**PrefixDialIn AdHoc H323ID: <S: 0, 50>** **PrefixDialIn AdHoc H323Alias: <E164: 0, 31>** **PrefixDialIn DID H323ID: <S: 0, 50>** **PrefixDialIn DID H323Alias: <E164: 0, 31>**
QoS	**QoS [1..2] Precedence Telephony Audio: <0/1/2/3/4/5/6/7/Auto/Off>** **QoS [1..2] Precedence Telephony Signalling: <0/1/2/3/4/5/6/7/Auto/Off>**

	QoS [1..2] Precedence VideoTelephony Audio: <0/1/2/3/4/5/6/7/Auto/Off> **QoS [1..2] Precedence VideoTelephony Signalling: <0/1/2/3/4/5/6/7/Auto/Off>** **QoS [1..2] Precedence VideoTelephony Video: <0/1/2/3/4/5/6/7/Auto/Off>** **QoS [1..2] Precedence VideoTelephony Data: <0/1/2/3/4/5/6/7/Auto/Off>** **QoS [1..2] Diffserv Telephony Audio: <0..63>** **QoS [1..2] Diffserv Telephony Signalling: <0..63>** **QoS [1..2] Diffserv VideoTelephony Audio: <0..63>** **QoS [1..2] Diffserv VideoTelephony Signalling: <0..63>** **QoS [1..2] Diffserv VideoTelephony Video: <0..63>** **QoS [1..2] Diffserv VideoTelephony Data: <0..63>** **QoS [1..2] Mode: <Precedence/Diffserv/Off>** **QoS [1..2] ToS: <MinDelay/MaxThrough/MaxReliable/MinCost/Off>**
RTP	**RTP MTU: <1200..1400>**
SerialInterfaceCard	**SerialInterfaceCard [1..6] Port [1..32] Callcontrol: <RS366Adtran/Dial_Manual>** **SerialInterfaceCard [1..6] Port [1..32] Clocking: <Dual/Single>** **SerialInterfaceCard [1..6] Port [1..32] DTRPulse: <On/Off>** **SerialInterfaceCard [1..6] Port [1..32] Bandwidth: <64/128/192/256/320/384/512/768/1152/1472/1536/1920>** **SerialInterfaceCard [1..6] Port [1..32] Restrict: <On/Off>** **SerialInterfaceCard [1..6] Port [1..32] Clocking: <Dual/Single/Internal>** **SerialInterfaceCard [1..6] Description: <S: 0, 16>** **SerialInterfaceCard [1..6] InternalBandwidth: <64/128/192/256/320/384/512/768/1152/1472/1536/1920>**
Session	**TimeOut: <0..65534>**
SingleNumberDialIn	**SingleNumberDialIn Active: <On/Off>**

	SingleNumberDialIn Numbers ISDN: <S: 0, 60> **SingleNumberDialIn Numbers IP: <S: 0, 60>** **SingleNumberDialIn Numbers H323Alias: <E164: 0, 31>** **SingleNumberDialIn Numbers H323ID: <S: 0, 50>** **SingleNumberDialIn Numbers SIP: <S: 0, 60>** **SingleNumberDialIn Password: <On/Off>** **SingleNumberDialIn NumberOfLoginTries: <1..10>** **SingleNumberDialIn WaitingRoomTimer: <1..60>**
SIP	**SIP Mode: <On/Off>** **SIP Interface Server Address: <S: 0, 255>** **SIP Interface Server Type: <Auto/Nortel/Microsoft/ Cisco/Alcatel/Experimental>** **SIP Interface Authentication UserName: <S: 0, 80>** **SIP Interface Authentication Password: <S: 0, 60>** **SIP Interface Transport Default: <TCP/UDP>**
SNMP	**SNMP Mode: <On/Off/ReadOnly/TrapsOnly>** **SNMP CommunityName: <S: 0, 16>** **SNMP SystemContact: <S: 0, 70>** **SNMP SystemLocation: <S: 0, 70>** **SNMP HostIPAddr [1..3]: <IPv4v6Addr: 0, 43>**
SSH	**SSH Mode: <On/Off>**
STUN	**STUN Server Address: <S: 0, 255>**
SystemClock	**SystemClock Port: <0..32>**
SystemUnit	**Name: <S: 0, 50>** **Password: <S: 0, 16>** **ResourceManagement: <On/Off>** **ResourceManagementPercent: <0..100>** **DefaultIpConference: <0..15>**

	TerminalSyncLossTimer: <5..90> **GatekeeperAdhocString: <S: 0, 10>** **TrafficShaping: <On/Off>** **OutbandDTMF: <On/Off>** **LanguageAdmin: <S: 0, 3>** **LanguageGraphic: <S: 0, 3>** **DefaultConferenceTemplate: <1..10>**
Telnet	**Telnet Mode: <On/Off>**

* The Gateway LoadLimit command will not be available if 1) MPS doesn't have GW option; 2) MPS have MCU and GW option.

3.2 directory.xml – xdirectory

LocalEntry	**LocalEntry [1..250] Name: <S: 0, 48>** **LocalEntry [1.. 250] Number: <S: 0, 60>** **LocalEntry [1.. 250] SecondNumber: <S: 0, 60>** **LocalEntry [1.. 250] SubAddress: <S: 0, 60>** **LocalEntry [1.. 250] CallRate: <1xh221/2xh221/64/128/192/256/320/384/512/768/1152/1472/1920/Tlph/H0/Max/Auto>** **LocalEntry [1.. 250] Restrict: <On/Off>** **LocalEntry [1.. 250] NetProfile: <1..7>** **LocalEntry [1.. 250] NetworkId: <1..32>** **LocalEntry [1.. 250] NetworkModule: <0..6>**
GroupEntry	**GroupEntry [1..16] Name: <S: 0, 48>** **GroupEntry [1..16] LocalEntryId [1..32]: <0..250>**

4 API - Commands

This section gives an overview of the supported system Commands.
All examples are presented using the standard XACLI format.

4.1 command.xml – xcommand

Boot	Command used to reboot the system.
	Parameters:
	None
	OK Result parameters:
	None
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand boot`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

CallFURBlock	Command used to block Fast updates to a call **Parameters:** • **Call(r): <1..176>** • **Value(r): <On/Off>** **OK Result parameters:** None **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not accepted by the system • **Description** Textual description of the cause code.
CallMoveToQCIF	Command used to move a call to QCIF **Parameters:** • **Call(r): <1..108>** **OK Result parameters:** None **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not accepted by the system • **Description** Textual description of the cause code.
CallMute	Command used to mute incoming audio from a specific call. **Parameters:** • **Call(r): <1..128>** Reference to the call to be muted or unmuted. • **Mode(r): <On/Off>** Denotes whether the call is to be muted or unmuted. **OK Result parameters:** None **ERROR Result parameters:**

	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand callmute call:2 mode:on`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

CallMuteOutgoing	Command used to mute outgoing audio from a specific call.
	Parameters:
	• **Call(r): <1..128>** Reference to the call to be muted or unmuted.
	• **Mode(r): <On/Off>** Denotes whether the call is to be muted or unmuted.
	OK Result parameters:
	None
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand CallMuteOutgoing call:2 mode:on`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

CallTransfer	Command used to transfer the H.323 party of a Gateway call to another H.323 party. **Parameters:** • GatewayCall(r): <1..80 • Number(r): <S: 0, 60> **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the call was not accepted by the system • **Description** Textual description of the cause code.
CallTransferToDirEntry	Command used to transfer the H.323 party of a Gateway call to a H.323 directory entry in the local phone book. **Parameters:** • GatewayCall(r): <1..80> • DirectoryEntry(r): <1..99> **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the call was not accepted by the system • **Description** Textual description of the cause code.

CallMuteVideo	Command used to mute incoming video from a specific call.
	Parameters:
	• **Call(r): <1..128>** Reference to the call to be muted or unmuted.
	• **Mode(r): <On/Off>** Denotes whether video is to be muted or unmuted.
	OK Result parameters:
	None
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand callmutevideo call:2 mode:on`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

ConferenceDisconnect	Command used to disconnect all calls in a conference.
	Parameters:
	• **Conference(r): <1..12>**
	OK Result parameters:
	None
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommmand conferencedisconnect conference:1`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

ConferenceModify	Command used to modify conference settings. **Parameters:** • **Conference(r): <1..35>** • **PictureMode: <Auto/VS/2Split/3Split/4Split/4+3Split/5+1Split/7+1Split/8+2SplitTopBottom/8+2SplitBottomTop/8+2SplitTop/8+2SplitBottom/9Split/12+1SplitCenter/12+1SplitTopLeft/12+2Split/16Split/30+2Split/2+1SplitWide/3SplitWide/3+1SplitWide/4SplitWide/4+1SplitWide/6SplitWide/8+1SplitWide/12SplitWide/CPAuto>** • **VideoFormat: <Auto/Motion/Sharpness>** • **CustomFormats: <On/Off>** • **AGC: <On/Off>** • **AllowIncomingCalls: <On/Off>** • **Duration: <0..999>** • **MaxAudioSites: <0..16>** • **MaxVideoSites: <0..160>**

	• **EntryExitTones: <On/Off>**
	• **LegacyLevel: <0..15>**
	• **TelephoneFilter: <On/Off>**
	• **FloorToFull: <On/Off>**
	• **BandwidthThreshold: <64/128/192/256/320/384/512/768/1152/1472/1920>**
	• **WebCallListTimeout: <On/Off>**
	• **PhoneIndication: <On/Off>**
	• **SpeakerIndication: <On/Off>**
	• **VideoText: <On/Off/Auto>**
	• **VideoTextTimeout: <0..30>**
	• **ChairControl: <On/Off>**
	• **LectureMode: <On/Off>**
	• **NetErrorHandling: <IPLR/FURBlock/None>**
	• **IPLRRobustMode: <Auto/On>**
	• **FURBlockSites: <Auto/On>**
	• **FURFilterInterval: <0..60>**
	• **VoiceSwitchTimeout: <1..10>**
	• **FarTlphEchoSupression: <Off/Normal/High>**
	• **OptimalVideoQuality: <On/Off>**
	• **EncoderSelectionPolicy: <BestBitRate/BestVideoStandard/BestResolution>**
	• **BandwidthManagement: <Manual/Auto>**
	• **WebSnapshots: <On/Off>**
	
	OK Result parameters:
	None
	
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	
	Example:
	`xcommand conferencemodify conference:1 agc:on floortofull:on`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

ConferenceRedefine	Command used to modify conference settings for a started conference with no active calls. **Parameters:** • **Conference(r): <1..35>** • **Name: <S: 0, 30>** • **CallRate: <1xh221/2xh221/64/128/192/256/320/384/H0/512/768/1152/1472/1536/1920/Tlph>** • **Restrict: <On/Off>** • **Password: <S: 0, 8>** • **PasswordOnOutgoingCalls: <On/Off>** • **Encryption: <On/Off>** • **EncryptionType: <DES/AES-128/Auto>** • **SecondaryRate: <On/Off>** • **WelcomeMessage: <On/Off>** • **DuoVideo: <On/Off>**

	• **AudioG728: <On/Off>**
	• **CascadingPreference: <Auto/Master/Slave>**
	• **BillingCode: <S: 0, 16>**
	• **CPAutoSwitch: <0..60>**
	• **NetworkId: <1..32>**
	• **ConferenceSelfview: <On/Off>**
	• **Protect: <On/Off>**
	• **TemplateReference: <1..10>**
	• **HDEnabled: <On/Off>**
	
	OK Result parameters:
	None
	
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	
	Example:
	`xcommand conferenceredifine conference:1 duovideo:on restrict:on`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

ConferenceStart	Command used to start a new conference. **Parameters:** • **Conference(r): <1..35>** • **Name: <S: 0, 30>** • **CallRate: <1xh221/2xh221/64/128/192/256/320/384/H0/512/768/1152/1472/1536/1920/Tlph>** • **Restrict: <On/Off>** • **Password: <S: 0, 8>** • **PasswordOnOutgoingCalls: <On/Off>** • **Encryption: <On/Off>** • **EncryptionType: <DES/AES-128/Auto>** • **SecondaryRate: <On/Off>** • **WelcomeMessage: <On/Off>** • **DuoVideo: <On/Off>** • **AudioG728: <On/Off>** • **CascadingPreference: <Auto/Master/Slave>** • **BillingCode: <S: 0, 16>** • **CPAutoSwitch: <0..60>** • **NetworkId: <1..32>** • **ConferenceSelfview: <On/Off>** • **Protect: <On/Off>** • **TemplateReference: <1..10>** • **HDEnabled: <On/Off>onference(r): <1..12>** **OK Result parameters:** None **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not accepted by the system

	• **Description** Textual description of the cause code.
	Example:
	`xcommand conferencestart conference:1`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

ConferenceStop	Command used to stop a conference. All active calls must be disconnected prior to stopping the conference.
	Parameters:
	• **Conference(r): <1..35>**
	OK Result parameters:
	None
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand conferencestop conference:1`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

ConferenceShowDuration	Command used to show remaining time of conference. **Parameters:** • **Conference(r): <1..15>** • **Time(r): <1Min/5Min/10Min/Off>** **OK Result parameters:** None **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the call was not accepted by the system • **Description** Textual description of the cause code.

ConferenceTerminate	Command used to terminate a conference.
	Parameters:
	• **Conference(r): <1..35>**
	OK Result parameters:
	None
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand ConferenceTerminate conference:1`
	
	`*r Result (status=OK):`

	`*r/end`
	
	`OK`

DefaultValuesSet	Command used to reset configurations to default values.
	
	Parameters:
	• **Level: <1..3>** Configurations are divided into three different storage classes. The level parameter denotes that configurations on this level and all levels below are to be reset.
	
	OK Result parameters:
	None
	
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	
	Example:
	`xcommand defaultvaluesset level:2`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

Dial	Command used to initiate an outgoing call.
	
	Parameters:
	• **Conference(r): <1..35>**
	• **Number: <S: 0, 60>** Number to dial.
	• **SecondNumber: <S: 0, 60>** 2Xh221 second number
	• **SubAddress: <S: 0, 60>** Sub address
	• **CallRate: <1xh221/2xh221/64/128/192/256/320/384/H0/512/768/1152/1472/1536/1920/Tlph>** Specifies the callrate to use
	• **Restrict: <On/Off>**
	• **NetProfile: <1..7>**
	• **NetworkId: <1..32>**
	• **NetworkModule: <1..12>**
	• **DTMFSend: <S: 0, 32>**
	
	OK Result parameters:
	• **CallRef: <1..96>** Reference to the call. To be used as reference when monitoring the call.
	• **LogTag: <1…>** Unique reference to call. Identifies the call in the call log.
	
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the call was not accepted by the system
	• **Description** Textual description of the cause code.
	
	Example:
	`xcommand dial number:666 callrate:256 netprofile:3`
	
	`*r Result (status=OK):`
	`  CallRef: 26`
	`  LogTag: 312`
	`*r/end`

	OK

DialGroupEntry	Command used to dial an entry from the Group Directory. Dialing from the Group Directory makes it possible to set up a MultiSite conference in one operation.
	Parameters:
	• **Conference(r): <1..35>**
	• **GroupEntryId(r): <1..16>** Reference to the directory entry to be dialed.
	OK Result parameters:
	The system will return the following elements for each call initiated.
	• **CallRef: <1..96>** Reference to the call. To be used as reference when monitoring the call.
	• **LogTag: <1…>** Unique reference to call. Identifies the call in the call log.
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the call was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand dialgroupentry conference:1 groupentryid:19`
	
	`*r Result (status=OK):`
	`CallRef: 2`
	`LogTag: 313`
	`CallRef: 1`
	`LogTag: 312`
	`CallRef:`
	`*r/end`
	
	`OK`

DialInNumberAdd	Command used to add a dial in number. **Parameters:** • **Conference(r): <1..15>** • **NetProfile(r): <2..7>** • **Number(r): <S: 0, 60>** • **CallerId: <S: 0, 60>** **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the call was not accepted by the system • **Description Textual description of the cause code.**
DialInNumberRemove	Command used to remove a dial in number. **Parameters:** • **Conference(r): <1..15>** • **NetProfile(r): <2..7>** • **Number(r): <S: 0, 60>** • **CallerId: <S: 0, 60>** **ERROR Result parameters:**

	• **Cause: <1…>** Cause code specifying why the call was not accepted by the system • **Description Textual description of the cause code.**

DialLocalEntry	Command used to dial a number from the locally stored directory.
	Parameters:
	• **Conference(r): <1..35>**
	• **LocalEntryId(r): <1..99>** Reference to the directory entry to be dialed.
	OK Result parameters:
	• **CallRef: <1..96>** Reference to the call. To be used as reference when monitoring the call.
	• **LogTag: <1…>** Unique reference to call. Identifies the call in the call log.
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the call was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand diallocalentry localentryid:15`
	
	`*r Result (status=OK):`
	`  CallRef: 1`
	`  LogTag: 312`
	`*r/end`
	
	`OK`

DisconnectCall	Command used to disconnect a call.
	Parameters:
	• **Call(r): <1..176>** Reference to the call to be disconnected.
	OK Result parameters:
	None
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand disconnectcall call:9`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

DisconnectGWCall	Command used to disconnect a gateway call. **Parameters:** • **GatewayCall(r): <1..80>** Reference to the call to be disconnected. **OK Result parameters:** None **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not

	accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand disconnectgwcall gatewaycall:1`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

FeedbackDeregister	Command used to deregister XML feedback over HTTP(S).
	Parameters:
	• **ID: <1..3>** ID for the registration to deregister.
	OK Result parameters:
	• **ID: <1..3>**
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand feedbackderegister id:1`
	
	`*r Result (status=OK):`
	`  ID: 2`
	`*r/end`
	
	`OK`

FeedbackRegister	Command used to instruct the system to return XML feedback over HTTP(S) to specific URLs. The parts of the Status and Configuration XML documents to monitor are specified by XPath expressions.
	The system supports issuing feedback to 3 different URLs.
	The system allows a total of 20 XPath expressions to be registered, with a maximum of 15 for a single URL.
	Parameters:
	• **ID: <1..3>** ID for the registration. If this parameter is omitted the system uses the first vacant ID.
	• **URL(r): <S: 0, 256>** The URL to post feedback to.
	• **Expression.1..15: <S: 0, 256>** XPath expression
	OK Result parameters:
	• **ID: <1..3>**
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand feedbackregister url:http://10.47.14.185:8000`
	`expression.1:status/call`
	`expression.2:status/conferenc`
	`e`
	
	`*r Result (status=OK):`
	`  ID: 2`

	`*r/end`
	
	`OK`

FloorToSite	Command used to assign floor to a specific site in a conference.
	Parameters:
	• **Conference(r): <1..35>**
	• **MCUID(r): <1..191>** MCUID to the MultiSite the site is connected to.
	• **TerminalID(r): <1..191>** The site's terminal id, referenced to the MultiSite it is connected to.
	OK Result parameters:
	None
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand floortosite conference:4 mcuid:85 terminalid:2`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

FloorToSiteEnd	Command used to end the assignment of floor to a specific site in a conference supporting. Requires that the command *FloorToSite* has been issued in advance.
	Parameters:
	• **Conference(r): <1..35>**
	OK Result parameters:
	None
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand floortositeend`
	
	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

GroupEntryAdd	Command used to add a new Group entry to the locally stored Group Directory (or MultiSite Directory). The entry is stored in the first vacant position in the Group Directory. **Parameters:** • **Name: <S: 0, 48>** The entry's name. • **LocalEntryId.1..32: <1..99>**References to local entry ids to be included in this Group entry. **OK Result parameters:**

	• **GroupEntryId: <1..16>** Reference to the Group Directory position the entry is stored.
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand groupentryadd name:"The Team" localentryid.1:17 localentryid.2:29 localentryid.3:56`
	`*r Result (status=OK): GroupEntryId: 15 *r/end`
	`OK`

GroupEntryDelete	Command used to delete an entry in the locally stored Group Directory.
	Parameters:
	• **GroupEntryId(r): <1..16>** Reference to the entry to delete.
	OK Result parameters:
	None
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	Example:
	`xcommand groupentrydelete groupentryid:15`
	`*r Result (status=OK): *r/end`
	OK

LayoutLock	Command use to lock a call id (e.g. a participant) to a given window in the CP layout. The numbering goes from left to right, starting with window number one at the top row. **Note**: If *ConferenceSelfview* is set to Off it is only possible to lock the call id to window number one. **Parameters:** • **Window(r): <1..16>** • **CallID(r): <1..176> Call.** **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not accepted by the system • **Description** Textual description of the cause code.
LayoutUnLock	Command use to unlock a locked call id (e.g. a participant) from a given window in the CP layout. The numbering goes from left to right, starting with window number one at the top row. **Note**: Both the window id and the call id must be defined. **Parameters:**

	• **Window(r): <1..16>** • **CallID(r): <1..176> Call.** **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not accepted by the system • **Description** Textual description of the cause code.

LocalEntryAdd	Command used to add a new entry to the locally stored Directory. The entry is stored in the first vacant position in the Directory.
	
	Parameters:
	• **Name: <S: 0, 48>** The entry's name.
	• **Number: <S: 0, 60>** The entry's number.
	• **SecondNumber: <S: 0, 60>** The entry's second number (2XH221 number).
	• **SubAddress: <S: 0, 60>** The entry's sub address.
	• **CallRate: <1xh221/2xh221/64/128/192/256/320/384/H0/512/768/1152/1472/1536/1920/Tlph>** The callrate to use when calling this entry.
	• **Restrict: <On/Off>** Whether to use restrict or not when calling this entry.
	• **NetProfile: <1..7>** The Net Profile to use when calling this entry.
	• **NetworkId: <1..32>**
	• **NetworkModule: <1..6>**
	
	OK Result parameters:
	• **LocalEntryId: <1..250>** Reference to the Directory position the entry is stored.
	
	ERROR Result parameters:
	• **Cause: <1…>** Cause code specifying why the command was not accepted by the system
	• **Description** Textual description of the cause code.
	
	Example:
	`xcommand localentryadd name:"John Galt" number:666`
	
	`*r Result (status=OK):`
	`  LocalEntryId: 17`
	`*r/end`
	
	`OK`

LocalEntryDelete	Command used to delete an entry in the locally stored Directory. **Parameters:** • **LocalEntryId(r): <1..99>** Reference to the entry to delete. **OK Result parameters:** None **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not accepted by the system • **Description** Textual description of the cause code. **Example:** `xcommand localentrydelete localentryid:66`

	`*r Result (status=OK):`
	`*r/end`
	
	`OK`

OptionKeyAdd	Command used to add option key. **Parameters:** • **Key(r): <S: 0, 90>** Option key string **OK Result parameters:** None **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not accepted by the system • **Description Textual description of the cause code.**
OptionKeyDelete	Command used to remove option key. **Parameters:** • **OptionKeyId(r): <1..64>** Option key Id to remove **OK Result parameters:** None **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not accepted by the system • **Description Textual description of the cause code.**
ParticipantMove	Command used to move participant from from one conference to another conference. **Parameters:** • **FromConferenceItem(r): <1..35> Conference to move from** • **CallItem(r): <1..176> Call to move** • **ToConferenceItem(r): <1..35> Conference to move to.** **OK Result parameters:** None **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not accepted by the system • **Description Textual description of the cause code.**
ProtectNumberAdd	Command used to add a protect number. **Parameters:** • **Conference(r): <1..15> Conference to add number** • **NetProfile(r): <2..7> Net profile of number** • **Number(r): <S: 0, 60>** Number to add **OK Result parameters:** None **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not accepted by the system • **Description Textual description of the cause code.**

ProtectNumberRemove	Command used to remove a protect number. **Parameters:** • **Conference(r): <1..15> Conference to remove number** • **NetProfile(r): <2..7> Net Profile to remove** • **Number(r): <S: 0, 60> Number to remove** **OK Result parameters:** None **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not accepted by the system • **Description Textual description of the cause code.**
VideoTextSet	Command used to set video text for a call. **Parameters:** • **CallItem(r): <1..176> Call to set text to** • **Text: <S: 0, 61> The text to set** • **Visible: <On/Off> If text is visible** **OK Result parameters:** None **ERROR Result parameters:** • **Cause: <1…>** Cause code specifying why the command was not accepted by the system • **Description Textual description of the cause code.**

5 API - Status

This section gives an overview of the Status Information available in the Status XML documents (status.xml / history.xml) and the Status root commands (xstatus / xhistory) of the XACLI interface.

All examples are presented using the standard XACLI format.

5.1 status.xml – xstatus

Call [1..160]

Top level attributes:

- **conferenceRef: 1..9**
- **direction: Incoming/Outgoing**
- **logTag: 1…** Unique number identifying the call. This tag can be used to track the call in the call log (*history.xml / xhistory*).
- **protocol: H320/H323**
- **status: CallIDLE/Dialing/Alerting/Proceeding/EstablOut/EstablIn/ AwaitInCnf/Connected/Disconnecting/Disconnected/Await2ndnr/ ClearOut/ClearIn/Syncing/Capex/Synced/Unframed**
- **type: Tlph/Vtlph**

Summary:

- Returns all currently available information for a call.

Examples:

```
*s Call 1 (status=Synced, type=Vtlph, protocol=H323,
direction=Outgoing, logTag=1, conferenceRef=1):
     CallRate: 384
     RemoteNumber: "5020020"
     Mute: Off
     Microphone: Off
     Duration: 16036
     PictureMode: Full
     RequestingPassword: Off
     MuteVideo: Off
     MuteOutgoing: Off
     VideoText: "System20020-NOR"
     FURBlock: Off
     FURAlert: Off
     LayoutLock: Off
     LayoutLockWindow: 0
     Channels 1 (type=Incoming):
       Rate: 384
       Restrict: Off
       Encryption (status=Off): /
       Audio (status=Active):
         Protocol: AAC-LD
         Rate: 64
         Level: 0
         RemoteIPAddress: ""
         LocalIPAddress: "10.47.9.150:2342"
         Encryption (status=Off): /
         RSVP: Off
         RSVPRate: 0
         DynamicRate: 64
         TotalPackets: 801748
         PacketLoss: 0
         Jitter: 0
       Video 1 (status=Active):
         Protocol: H264
         Resolution: SIF
         Rate: 160
         RemoteIPAddress: ""
```

```
          LocalIPAddress: "10.47.9.150:2344"
          Encryption (status=Off): /
          RSVP: Off
          RSVPRate: 0
          DynamicRate: 157
          TotalPackets: 476674
          PacketLoss: 1
          Jitter: 10
        Video 2 (status=Active):
          Protocol: H263+
          Resolution: XGA
          Rate: 160
          RemoteIPAddress: ""
          LocalIPAddress: "10.47.9.150:2346"
          Encryption (status=Off): /
          RSVP: Off
          RSVPRate: 0
          DynamicRate: 72
          TotalPackets: 161681
          PacketLoss: 0
          Jitter: 27
        Data (status=Inactive): /
      Channels 2 (type=Outgoing):
        Rate: 384
        Restrict: Off
        Encryption (status=Off): /
        Audio (status=Active):
          Protocol: AAC-LD
          Rate: 64
          RemoteIPAddress: "10.47.20.20:2334"
          LocalIPAddress: "10.47.9.150:2342"
          Encryption (status=Off): /
          RSVP: Off
          RSVPRate: 0
          DynamicRate: 64
          TotalPackets: 801652
          PacketLoss: 1
          Jitter: 0
        Video 1 (status=Active):
          Protocol: H264
          Resolution: CIF
          Rate: 320
          RemoteIPAddress: "10.47.20.20:2336"
          LocalIPAddress: "10.47.9.150:2344"
          Encryption (status=Off): /
          RSVP: Off
          RSVPRate: 0
          DynamicRate: 164
          TotalPackets: 259599
          PacketLoss: 0
          Jitter: 7
        Video 2 (status=Inactive): /
        Data (status=Inactive): /
*s/end

*s Call 2 (status=Disconnected, type=NA,
protocol=NA, direction=NA, logTag=NA,
conferenceRef=NA):
     Cause: 255
*s/end
```

Conference [1..40]

Top level attributes:

- **status: NotStarted/Started/Active**

Summary:

- Includes references to the calls being connected to the conference
- DuoVideo status
- Includes information about the pictures generated by the MultiSite
- Cascading status
- MCU Site list
- On Air information

Examples:

```
*s Conference 1 (status=NotStarted): /
*s/end

*s Conference 1 (status=Started):
     MCUID: 85
     Properties:
       Name: "Marius sin"
       CallRate: 384
       Restrict: Off
       Password: ""
       PasswordOnOutgoingCalls: Off
       Dynamic: On
       ID: ""
       Encryption: Off
       EncryptionType: Auto
       WelcomeMessage: On
       DuoVideo: On
       AudioG728: On
       CascadingPreference: Auto
       BillingCode: ""
       CPAutoSwitch: 0
       PictureMode: VS
       VideoFormat: Auto
       CustomFormats: On
       AGC: On
       AllowIncomingCalls: On
       Duration: 0
       MaxAudioSites: 0
       MaxVideoSites: 0
       EntryExitTones: On
       LegacyLevel: 0
       TelephoneFilter: On
       FloorToFull: On
       WebCallListTimeout: On
       BandwidthThreshold: 128
       NetworkId: 1
       ConferenceSelfview: On
       PhoneIndication: On
       SpeakerIndication: On
       VideoText: On
       VideoTextTimeout: 2
       ChairControl: On
       LectureMode: Off
       Protect: Off
       VideoCodingMode: Transcoding
       SecondaryRate: On
```

```
        FarTlphEchoSupression: Normal
        NetErrorHandling: None
        IPLRRobustMode: Auto
        FURBlockSites: Auto
        FURFilterInterval: 3
        HDEnabled: Off
        VoiceSwitchTimeout: 2
        OptimalVideoQuality: On/Off
        EncoderSelectionPolicy:
          BestBitRate/BestVideoStandard
        BandwidthManagement: Auto/Manual
        WebSnapshots: On/Off
        AutoAspectRatio4x3: Off
*s/end

*s Conference 1 (status=Active):
     Calls:
       CallRef 1: 1
       CallRef 2: 2
       CallRef 3: 3
     DuoVideo (status=Off): /
     Floor: None
     Current:
       CallRef: 3
     Previous:
       CallRef: 2
     OutgoingPicture 1 (name=Current):
       Layout (type=5+1Split):
         Window 1:
           Picture: RemoteMain
           CallRef: 3
         Window 2:
           Picture: RemoteMain
           CallRef: 2
         Window 3:
           Picture: RemoteMain
           CallRef: 1
         Window 4:
           Picture: NA
           CallRef: None
         Window 5:
           Picture: NA
           CallRef: None
         Window 6:
           Picture: NA
           CallRef: None
     OutgoingPicture 2 (name=Previous):
       Layout (type=5+1Split):
         Window 1:
           Picture: RemoteMain
           CallRef: 2
         Window 2:
           Picture: RemoteMain
           CallRef: 3
         Window 3:
           Picture: RemoteMain
           CallRef: 1
         Window 4:
           Picture: NA
           CallRef: None
         Window 5:
```

```
        Picture: NA
        CallRef: None
      Window 6:
        Picture: NA
        CallRef: None
  OutgoingPicture 3 (name=Duo):
    Layout (type=NA): /
  PictureModeActual: Full
  MCUID: 1
  CascadingMode: StandAlone
  MCUSiteList:
    Site 1:
      MCUID: 1
      TerminalID: 2
      Name: "System1"
      CallRef: 1
    Site 2:
      MCUID: 1
      TerminalID: 3
      Name: "System2"
      CallRef: 2
    Site 3:
      MCUID: 1
      TerminalID: 4
      Name: "System3"
      CallRef: 3
  IPOnly: On
  FullySwitched: Off
  Dynamic: Off
  Hybrid: On
  Properties:
    Name: "TCS Test"
    CallRate: 384
    Restrict: Off
    Password: ""
    PasswordOnOutgoingCalls: Off
    Dynamic: Off
    Encryption: Off
    EncryptionType: Auto
    WelcomeMessage: On
    DuoVideo: On
    AudioG728: On
    CascadingPreference: Auto
    BillingCode: ""
    CPAutoSwitch: 0
    PictureMode: 4Split
    VideoFormat: Auto
    CustomFormats: On
    AGC: On
    AllowIncomingCalls: On
    Duration: 0
    MaxAudioSites: 8
    MaxVideoSites: 40
    EntryExitTones: On
    LegacyLevel: 0
    TelephoneFilter: On
    FloorToFull: On
    WebCallListTimeout: Off
    BandwidthThreshold: 128
    NetworkId: 1
    ConferenceSelfview: On
```

```
        PhoneIndication: On
        SpeakerIndication: On
        VideoText: On
        VideoTextTimeout: 5
        ChairControl: Off
        LectureMode: Off
        Protect: Off
        VideoCodingMode: Transcoding
        SecondaryRate: Off
        FarTlphEchoSupression: Normal
        NetErrorHandling: None
        IPLRRobustMode: Auto
        FURBlockSites: Auto
        FURFilterInterval: 3
        HDEnabled: Off
        VoiceSwitchTimeout: 2
*s/end
```

Ethernet [1..2]

Top level attributes:
None

Summary:
- Returns MAC Address
- Returns Ethernet speed

Example

```
*s Ethernet 1:
     MacAddress: "00:50:60:7F:FA:FB"
     Speed: 10full
*s/end
```

Feedback [1..3]

Top level attributes:
- **status: on/off** Indicates if there are HTTP feedback registered for a given Feedback ID, ref. command *FeedbackRegister*.

Summary:
- Lists the URL and feedback expressions registered for the given Feedback ID

Examples:

```
*s Feedback 1 (status=Off): /
*s/end

*s Feedback 1 (status=On):
     URL: "http://10.47.14.185:8000/"
     Expression: "status/call[@status="Synced"]"
     Expression: ""
     Expression: ""
     Expression: ""
     Expression: ""
     Expression: ""
     Expression: ""
     Expression: ""
     Expression: ""
     Expression: ""
     Expression: ""
     Expression: ""
     Expression: ""
     Expression: ""
     Expression: ""
     Expression: ""
```

	`*s/end`

H323Gatekeeper [1..2]	**Top level attributes:**
	• **status:Required/Discovering/Discovered/Authenticating/ Authenticated/Registering/Registered/Rejected/Inactive**
	Summary:
	• Returns H323Gatekeeper status
	Examples:
	`*s H323Gatekeeper 1 (status=Inactive): /`
	`*s/end`
	
	`*s H323Gatekeeper 1 (status=Registered):`
	`  Address: "10.47.9.1"`
	`  Port: 1719`
	`*s/end`
	
	`*s H323Gatekeeper 1 (status=Rejected):`
	`  Address: "10.47.9.0"`
	`  Port: 0`
	`*s/end`

IP [1..2]	**Top level attributes:**
	None
	Summary:
	• Returns current IP address, Subnet Mask and Gateway address
	Example
	`*s IP 1:`
	`  Address: "10.47.8.222"`
	`  SubnetMask: "255.255.248.0"`
	`  Description: ""`
	`  Gateway: "10.47.8.1"`
	`  V6:`
	`  Address 1 (type=NA): ""`
	`  Address 2 (type=NA): ""`
	`  DNS:`
	`  Server 1:`
	`  Address: "10.0.0.2"`
	`  Server 2:`
	`  Address: "127.0.0.1"`
	`  Server 3:`
	`  Address: "127.0.0.1"`
	`  Server 4:`
	`  Address: "127.0.0.1"`
	`  Server 5:`
	`  Address: "0.0.0.0"`
	`  Domain:`
	`  Name: ""`
	`*s/end`

MediaBoard [1..8]	**Top level attributes:** None **Summary:** • Returns current IP and Ethernet information for the Media Boards.

	Example
	`*s MediaBoard 1:`
	`  TemperatureCelcius: 30`
	`  TemperatureFahrenheit: 86`
	`  Description: ""  IP:`
	`  Address: "10.47.9.106"`
	`  SubnetMask: "255.255.248.0"`
	`  Gateway: "10.47.8.1"`
	`  V6:`
	`  Address 1 (type=NA): ""`
	`  Address 2 (type=NA): ""`
	`  Ethernet:`
	`  MacAddress: "00:50:60:00:ED:1F"`
	`  Speed: 100full`
	`*s/end`

SystemUnit	**Top level attributes:**
	None
	Summary:
	• Returns information about the System Unit
	Example
	`*s SystemUnit:`
	`  ProductType: "TANDBERG MPS-MCU"`
	`  ProductId: "TANDBERG MPS200"`
	`  Uptime: 600587`
	`  UptimeStr: "6 days 22 hours 49 minutes 47 seconds"`
	`  Software:`
	`  Version: "J3.0Beta9 (TEST SW)"`
	`  Name: "test"`
	`  ReleaseDate: "2006-01-13, 17:44, ert"`
	`  Configuration:`
	`  Telephony: 16`
	`  VideoTelephony: 40`
	`  AdvancedVideoOption: 40`
	`  Encryption: 40`
	`  PRIPorts: 0`
	`  SerialPorts: 0`
	`  MaxBW: 30720`
	`  ManagedResources:`
	`  Conference: 0`
	`  Telephony: 0`
	`  VideoTelephony: 0`
	`  AdvancedVideoOption: 0`
	`  Encryption: 0`
	`  MaxBW: 0`
	`  MaxBChanPRI: 0`
	`  TotalMCUResources:`
	`  Conference: 10`
	`  Telephony: 16`
	`  VideoTelephony: 40`
	`  MaxBW: 30720`
	`  MaxBChanPRI: 0`
	`  TotalGWResources:`
	`  GatewayCall: 0`
	`  MaxBChanPRI: 0`
	`  Hardware:`

	SerialNumber: "44a00003" MainBoard: "113637 MCP 820 System Controller" BootSoftware: "PPCBUG" Used: Total: Conference: 2 Telephony: 0 VideoTelephony: 1 BChanPRI: 0 Bandwidth: 384 Gateway: GatewayCall: 0 BChanPRI: 0 MCU: Conference: 2 Telephony: 0 VideoTelephony: 1 Bandwidth: 384 BChanPRI: 0 *s/end
SerialInterfaceCard	**Top level attributes:** • **status: on/off** **Summary:** • Returns information about the Serial interface card **Example** *s SerialInterfaceCard 1 (status=Off): Description: "" *s/end
ISDNInterfaceCard	*s ISDNInterfaceCard 1 (status=On): Description: "" PRI 1 (ready=True): BChannelsTotal: 29 BChannelsFree: 28 H0ChannelsFree: 4 Channels 1 (type=BChannel, status=Idle): / Channels 2 (type=BChannel, status=Idle): / Channels 3 (type=BChannel, status=Idle): / Channels 4 (type=BChannel, status=Idle): / Channels 5 (type=BChannel, status=Idle): / Channels 6 (type=BChannel, status=Idle): / Channels 7 (type=BChannel, status=Idle): / Channels 8 (type=BChannel, status=Idle): / Channels 9 (type=BChannel, status=Idle): / Channels 10 (type=BChannel, status=Idle): / Channels 11 (type=BChannel, status=Idle): / Channels 12 (type=BChannel, status=Idle): / Channels 13 (type=BChannel, status=Idle): / Channels 14 (type=BChannel, status=Idle): / Channels 15 (type=BChannel, status=Idle): / Channels 16 (type=DChannel, status=NA): / Channels 17 (type=BChannel, status=Idle): / Channels 18 (type=BChannel, status=Idle): / Channels 19 (type=BChannel, status=Idle): / Channels 20 (type=BChannel, status=Idle): / Channels 21 (type=BChannel, status=Idle): /

```
       Channels 22 (type=BChannel, status=Idle): /
       Channels 23 (type=BChannel, status=Idle): /
       Channels 24 (type=BChannel, status=Idle): /
       Channels 25 (type=BChannel, status=Idle): /
       Channels 26 (type=BChannel, status=Idle): /
       Channels 27 (type=BChannel, status=Idle): /
       Channels 28 (type=BChannel,
status=Disconnected):
         CallingNumber: "098228199"
         CauseLocation: 0
         ChannelCause: 16
         ConnectionTime: 16
       Channels 29 (type=BChannel,
status=Disconnected):
         CallingNumber: "074859215"
         CauseLocation: 2
         ChannelCause: 16
         ConnectionTime: 11
       Channels 30 (type=BChannel,
status=Disconnecting):
         CallingNumber: "041669426"
         CauseLocation: 2
         ChannelCause: 16
         ConnectionTime: 0
       Channels 31 (type=BChannel,
status=Disconnected):
         CallingNumber: "71269140"
         CauseLocation: 0
         ChannelCause: 16
         ConnectionTime: 0
     PRI 2 (ready=True):
       BChannelsTotal: 29
       BChannelsFree: 29
       H0ChannelsFree: 4
       Channels 1 (type=BChannel, status=Idle): /
       Channels 2 (type=BChannel, status=Idle): /
       Channels 3 (type=BChannel, status=Idle): /
       Channels 4 (type=BChannel, status=Idle): /
       Channels 5 (type=BChannel, status=Idle): /
       Channels 6 (type=BChannel, status=Idle): /
       Channels 7 (type=BChannel, status=Idle): /
       Channels 8 (type=BChannel, status=Idle): /
       Channels 9 (type=BChannel, status=Idle): /
       Channels 10 (type=BChannel, status=Idle): /
       Channels 11 (type=BChannel, status=Idle): /
       Channels 12 (type=BChannel, status=Idle): /
       Channels 13 (type=BChannel, status=Idle): /
       Channels 14 (type=BChannel, status=Idle): /
       Channels 15 (type=BChannel, status=Idle): /
       Channels 16 (type=DChannel, status=NA): /
       Channels 17 (type=BChannel, status=Idle): /
       Channels 18 (type=BChannel, status=Idle): /
       Channels 19 (type=BChannel, status=Idle): /
       Channels 20 (type=BChannel, status=Idle): /
       Channels 21 (type=BChannel, status=Idle): /
       Channels 22 (type=BChannel, status=Idle): /
       Channels 23 (type=BChannel, status=Idle): /
       Channels 24 (type=BChannel, status=Idle): /
       Channels 25 (type=BChannel,
status=Disconnected):
         CallingNumber: "061336971"
```

```
          CauseLocation: 2
          ChannelCause: 17
          ConnectionTime: 0
        Channels 26 (type=BChannel,
status=Disconnected):
          CallingNumber: "061336971"
          CauseLocation: 2
          ChannelCause: 17
          ConnectionTime: 0
        Channels 27 (type=BChannel,
status=Disconnected):
          CallingNumber: "061336971"
          CauseLocation: 2
          ChannelCause: 17
          ConnectionTime: 0
        Channels 28 (type=BChannel,
status=Disconnected):
          CallingNumber: "061336971"
          CauseLocation: 0
          ChannelCause: 16
          ConnectionTime: 0
        Channels 29 (type=BChannel,
status=Disconnected):
          CallingNumber: "032875677"
          CauseLocation: 2
          ChannelCause: 17
          ConnectionTime: 0
        Channels 30 (type=BChannel,
status=Disconnected):
          CallingNumber: "032875677"
          CauseLocation: 2
          ChannelCause: 17
          ConnectionTime: 0
        Channels 31 (type=BChannel, status=Idle): /
      PRI 3 (ready=False):
        State: RedAlarm
      PRI 4 (ready=False):
        State: RedAlarm
      PRI 5 (ready=False):
        State: RedAlarm
      PRI 6 (ready=False):
        State: RedAlarm
      PRI 7 (ready=False):
        State: RedAlarm
      PRI 8 (ready=False):
        State: RedAlarm
      G703 1 (ready=False):
        State: RedAlarm
      G703 2 (ready=False):
        State: RedAlarm
      G703 3 (ready=False):
        State: RedAlarm
      G703 4 (ready=False):
        State: RedAlarm
      G703 5 (ready=False):
        State: RedAlarm
      G703 6 (ready=False):
        State: RedAlarm
      G703 7 (ready=False):
        State: RedAlarm
      G703 8 (ready=False):
```

	State: RedAlarm *s/end
ExternalManager	*s ExternalManager: Address: "10.47.1.10" Protocol: URL: "tms/public/external/management/SystemManagementService.asmx" *s/end
NTP	*s NTP: Address: "127.0.0.1" *s/end
SIP	*s SIP: Mode: On Interface: Server (status=Active): Address: "10.47.8.88" Authentication: Off Registration 1 (status=Inactive): URI: "" Registration 2 (status=Inactive): URI: "" Registration 3 (status=Inactive): URI: "" Registration 4 (status=Inactive): URI: "" Registration 5 (status=Inactive): URI: "" Registration 6 (status=Inactive): URI: "" Registration 7 (status=Inactive): URI: "" Registration 8 (status=Inactive): URI: "" Registration 9 (status=Inactive): URI: "" Registration 10 (status=Inactive): URI: "" Registration 11 (status=Inactive): URI: "" Registration 12 (status=Inactive): URI: "" Registration 13 (status=Inactive): URI: "" Registration 14 (status=Inactive): URI: "" Registration 15 (status=Inactive): URI: "" Registration 16 (status=Inactive): URI: "" Registration 17 (status=Inactive): URI: "" Registration 18 (status=Inactive): URI: "" Registration 19 (status=Inactive): URI: "" Registration 20 (status=Inactive):

	URI: "" Registration 21 (status=Inactive): URI: "" Registration 22 (status=Inactive): URI: "" Registration 23 (status=Inactive): URI: "" Registration 24 (status=Inactive): URI: "" Registration 25 (status=Inactive): URI: "" Registration 26 (status=Inactive): URI: "" Registration 27 (status=Inactive): URI: "" Registration 28 (status=Inactive): URI: "" Registration 29 (status=Inactive): URI: "" Registration 30 (status=Inactive): URI: "" Registration 31 (status=Inactive): URI: "" Registration 32 (status=Inactive): URI: "" Registration 33 (status=Inactive): URI: "" Registration 34 (status=Inactive): URI: "" Registration 35 (status=Inactive): URI: "" Registration 36 (status=Inactive): URI: "" Registration 37 (status=Inactive): URI: "" Registration 38 (status=Inactive): URI: "" Registration 39 (status=Inactive): URI: "" Registration 40 (status=Inactive): URI: "" *s/end
SystemLoad	*s SystemLoad: 2 *s/end
SystemClock	*s SystemClock: Port: Configuration: 0 Used: 1 Rate: 2048 State (status=OK): Info: PRI *s/end
SystemActivity	*s SystemActivity: CallsActive: 2 *s/end

Options	*s Options: Option 1: Key: "****" Description: "40 video telephony ports + 16 telephony ports." Option 2: Key: "****" Description: "Advanced Video Option on 40 sites." Option 3: Key: "****" Description: "Encryption on 40 sites." Option 4: / Option 5: / Option 6: / Option 7: / Option 8: / Option 9: / Option 10: / Option 11: / Option 12: / Option 13: / Option 14: / Option 15: / Option 16: / Option 17: / Option 18: / Option 19: / Option 20: / Option 21: / Option 22: / Option 23: / Option 24: / Option 25: / Option 26: / Option 27: / Option 28: / Option 29: / Option 30: / Option 31: / Option 32: / Option 33: / Option 34: / Option 35: / Option 36: / Option 37: / Option 38: / Option 39: / Option 40: / Option 41: / Option 42: / Option 43: / Option 44: / Option 45: / Option 46: / Option 47: / Option 48: / Option 49: / Option 50: / Option 51: /

```
     Option 52: /
     Option 53: /
     Option 54: /
     Option 55: /
     Option 56: /
     Option 57: /
     Option 58: /
     Option 59: /
     Option 60: /
     Option 61: /
     Option 62: /
     Option 63: /
     Option 64: /
*s/end
```

5.2 history.xml – xhistory

Call [1..96]	**Top level attributes:**
	• **type: Tlph/Vtlph**
	• **protocol: H320/H323/SIP**
	• **direction: Incoming/Outgoing**
	• **PartOf: Conference/GatewayCall**
	Summary:
	• Returns information about disconnected calls
	Examples:
	`*l Call 1 (type=Vtlph, protocol=H323, direction=Outgoing, PartOf=Conference):`
	`LogTag: 4`
	`ConferenceLogTag: 2`
	`ConferenceRef: 1`
	`RemoteNumber: "10.47.12.242"`
	`EncryptionIn: Off`
	`EncryptionOut: Off`
	`CallRate: 384`
	`DisconnectCauseValue: 16`
	`DisconnectCause: NA`
	`Duration: 0`
	`UptimeAtEndOfCall: 7758`
	`BillingCode: ""`
	`*l/end`

	Conference [1..20]
	Top level attributes:
	None
	Summary:
	• Returns information about ended conferences
	Examples:
	`*l Conference 1:`
	`LogTag: 1`
	`Name: ""`
	`CallRate: 384`
	`Restrict: Off`
	`Password: ""`
	`Encryption: Off`
	`Duration: 0`
	`*l/end`

5.3 Event.xml – xevent

AuthenticationFailure	***e AuthenticationFailure:** **Service: /** **RemoteIPAddress: /** **Uptime: /** ***e/end**
CallFURErrorAlert	***e CallFURErrorAlert:** **FURAlert: /** **Rate: /** **CallRef: /** **LogTag: /** **ConferenceRef: /** **ConferenceLogTag: /** ***e/end**
CallStatisticsIP	***e CallStatisticsIP:** **CallRef: /** **LogTag: /** **ConferenceRef: /** **ConferenceLogTag: /** **Duration: /** **Incoming:** **Audio:** **TotalPackets: /** **PacketLoss: /** **PacketsDropped: /** **MaxJitter: /** **Video:** **TotalPackets: /** **PacketLoss: /** **PacketsDropped: /** **MaxJitter: /** ***e/end**
CallSuccessful	***e CallSuccessful:** **CallRef: /**

	LogTag: / **ConferenceRef: /** **ConferenceLogTag: /** **Protocol: /** **Direction: /** **CallRate: /** **RemoteNumber: /** **Encryption:** **Incoming: /** **Outgoing: /** ***e/end**
DialInError	***e DialInError:** **IncomingNumber: /** **RemoteNumber: /** ***e/end**
DownspeedingFinished	***e DownspeedingFinished:** **CallRef: /** **LogTag: /** **ConferenceRef: /** **ConferenceLogTag: /** **Rate: /** ***e/end**
H320Statistics	***e H320Statistics:** **CallRef: /** **LogTag: /** **ConferenceRef: /** **ConferenceLogTag: /** **Duration: /** **FramingLoss:** **Last: /** **Total: /** **BondingReSync:** **Last: /** **Total: /** ***e/end**

SystemActivity	***e SystemActivity:** **Service: /** **RemoteIPAddress: /** **Uptime: /** **Description: /** ***e/end**

Filename: D1363906_MPS_API_Guide.doc
Directory: E:\margareth\Documents\TANDBERG DOCS\J41 AdminGuides\J41_MPS_APIGuide
Template: C:\Documents and Settings\m.wennersteen\Application Data\Microsoft\Templates\Normal.dot
Title: Updated for J4.x
Subject: MPS API Guide
Author:
Keywords:
Comments:
Creation Date: 11/16/2007 9:16:00 AM
Change Number: 5
Last Saved On: 11/29/2007 10:23:00 AM
Last Saved By: TANDBERG
Total Editing Time: 55 Minutes
Last Printed On: 11/29/2007 10:26:00 AM
As of Last Complete Printing
Number of Pages: 85
Number of Words: 14 260 (approx.)
Number of Characters: 94 976 (approx.)